Package ‘GeneGeneInteR’

April 25, 2017

Type Package

Title Tools for Testing Gene-Gene Interaction at the Gene Level

Version 1.2.0

Author Mathieu Emily, Nicolas Sounac, Florian Kroell, Magalie Houee-Bigot

Maintainer Mathieu Emily <mathieu.emily@agrocampus-ouest.fr>, Magalie Houee-Bigot <magalie.houee@agrocampus-ouest.fr>

Description The aim of this package is to propose several methods for testing gene-gene interaction in case-control association studies. Such a test can be done by aggregating SNP-SNP interaction tests performed at the SNP level (SSI) or by using gene-gene multidimensional methods (GGI) methods. The package also proposes tools for a graphic display of the results.

License GPL (>= 2)

LazyData TRUE

Depends R (>= 3.3)

biocViews GenomeWideAssociation, SNP, Genetics, GeneticVariability

Imports snpStats, mvtnorm, GGtools, Rsamtools, igraph, kernlab, FactoMineR, plspm, IRanges, GenomicRanges, data.table,rioja,grDevices, graphics,stats,utils

NeedsCompilation no

R topics documented:

CCA.test ......................................................... 2
CLD.test .......................................................... 3
data.SNP .......................................................... 4
gates.test ......................................................... 5
GBIGM.test ......................................................... 7
gene.pair .......................................................... 8
GGI ................................................................. 9
importFile ......................................................... 12
imputeSnpMatrix ............................................... 13
KCCA.test ........................................................ 15
minPtest .......................................................... 17
PCA.test .......................................................... 19
plot.GGInetwork .................................................. 20
PLSPM.test ....................................................... 23
print.GGItest ...................................................... 24
CCA.test

CCA (Canonical-Correlation Analysis) based GGI analysis.

Description
CCA.test performs a Gene-Gene Interaction (GGI) analysis based on the difference of canonical correlation between cases and controls.

Usage
CCA.test(Y, G1, G2, n.boot = 500)

Arguments

Y numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).

G1 SnpMatrix object. Must have a number of rows equal to the length of Y.

G2 SnpMatrix object. Must have a number of rows equal to the length of Y.

n.boot positive integer. n.boot is the number of bootstrap replicates for estimating variances. By default, this is fixed to 500.

Details
The test statistic is based on the difference between Fisher’s transformation of the maximum of the canonical correlations in cases and controls. To calculate the test statistic for the interaction p-value, CCA.test estimates the variance of the Fisher’s transformation of the maximum of the canonical correlations in cases and controls using a bootstrap method.

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

statistic The value of the statistic CCU.
p.value The p-value for the test.
estimate A vector of the Fisher’s transformed maximum canonical correlation coefficient in cases and controls.
parameter The number of bootstrap samples used to estimate the p-value.
null.value The value of CCU under the null hypothesis.
alternative a character string describing the alternative.
method a character string indicating the method used.
data.name a character string giving the names of the data.
CLD.test

References

See Also
GGI, KCCA.test

Examples

data(gene.pair)
CCA.test(Y=gene.pair$Y, G1=gene.pair$G1, G2=gene.pair$G2)

CLD.test

CLD (Composite Linkage Disequilibrium) based GGI analysis.

Description

CLD.test performs a Gene-Gene Interaction (GGI) analysis based on the difference between the Composite Linkage Disequilibrium measured in cases and controls respectively.

Usage

CLD.test(Y, G1, G2, n.perm = 1000)

Arguments

Y numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).

G1 SnpMatrix object. Must have a number of rows equal to the length of Y.

G2 SnpMatrix object. Must have a number of rows equal to the length of Y.

n.perm positive integer. n.perm is the number of permutations performed to compute the pvalue. By default, n.perm is fixed to 1000.

Details

The test statistic is based on Nagao normalized Quadratic Distance (NQD), as described in Rajapakse et al. (2012). The pvalue is calculated using n.perm permutations of Y.

Value

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

statistic The value of the statistic CLD.
p.value The p-value for the test.
estimate The estimation of CLD
parameter The number of permutations used to estimate the p-value.
null.value  The value of CLD under the null hypothesis.
alternative  a character string describing the alternative.
method       a character string indicating the method used.
data.name     a character string giving the names of the data.

References

See Also
GGI

Examples
data(gene.pair)
CLD.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2)

---

Multiple genes dataset

Description
A case-control dataset containing the genotypes of 312 SNPs from 17 genes in a total of 429 patients (266 individuals affected by Rheumatoid Arthritis and 163 Health controls)

Usage
data(data.SNP)

Format
A list with 3 objects:

- **snpX**: SnpMatrix object with 312 SNPs and 429 individuals.
- **genes.info**: a data frame where each SNP is described by its rs ID, its position and the gene it belongs to.
- **Y**: Factor vector of length 429 with 2 levels ("Health Control or "Rheumatoid"). Y corresponds to the case-control status response variable.

Value
A dataset containing the genotypes of 312 SNPs from 17 genes in a total of 429 patients.

Source
All three objects were taken from NCBI web site and are part of the GSE39428 series Chang X., et al. Investigating a pathogenic role for TXNDC5 in tumors. Int. J. Oncol., 43(6): 1871-84, 2013.
Description

`gates.test`, `minP.test`, `tTS.test` and `tProd.test` aim at performing gene-gene interaction analysis based on SNP-SNP interaction tests. The following methods are used to combine SNP-SNP interaction tests into a single Gene-Gene Interaction p-value:

- Minimum p-value in `minP.test` function
- Gene Association Test with Extended Simes procedure in `gates.test`
- Truncated Tail Strength in `tTS.test` function
- Truncated p-value Product in `tProd.test` function

Usage

`gates.test(Y, G1, G2, alpha = 0.05, me.est = c("ChevNy", "Keff", "LiJi", "Galwey"))`

Arguments

- `Y`: numeric or factor vector with exactly two different values. `Y` is the response variable and should be of length equal to the number of rows of `G1` and `G2` arguments (number of individuals).
- `G1`: `SnpMatrix` object. Must have a number of rows equal to the length of `Y`.
- `G2`: `SnpMatrix` object. Must have a number of rows equal to the length of `Y`.
- `alpha`: numeric value in [0, 1]. Threshold for GATES method when estimating the number of effective tests `Me` with `Keff` method.
- `me.est`: character string for GATES method. `me.est` corresponds to the method for estimating the number of effective tests. Must be one of the following: "ChevNy", "Keff", "LiJi", "Galwey" (See details).

Details

In a first step, all methods start by applying a logistic regression model to test all pairs of SNPs between the two genes `G1` and `G2`. If `G1` has `m1` SNPs and `G2` `m2` SNPs, a total of `m1 * m2` SNP-SNP tests are performed. In a second step, the `m1 * m2` SNP-SNP tests are combined according to their covariance matrix `Σ`. `Σ` is computed as described in the method developed in Emily (2016). The covariance `Σ` is used in each method as follows:

- minP test - minP test considered the significant of the observed minimum p-value. Significance is computed by integrating the multivariate normal distribution with covariance `Σ` as proposed in Conneelly and Boehnke (2008).
- GATES test - The p-value for GATES is the minimum p-value obtained after a multiple testing correction of the SNP-SNP interaction p-values. Correction for multiple testing is defined as

\[
me * p[i]/me[i]
\]

where `me` is the effective number of independent tests, `p[i]` is the i-th top p-values and `me[i]` is the effective number of independent test among the top i p-values. Many methods exist to estimate `me` and `me[i]` terms:
– Cheverud-Nyholt method (Cheverud, 2001 and Nyholt, 2004)
– Keff method (Moskovina and Schmidt, 2008)
– Li & Ji method (Li and Ji, 2005)
– Galwey method (Galwey, 2009)

Details of each method can be found in the references.

• tTS test - tTS test does not consider only the strongest signal but all signals that are inferior to a given threshold \( \tau \). For these p-values, the weighted sum of

\[
tTS = \sum (1 - p[i] * (m1 * m2 + 1)) / i
\]

is computed and represents the test statistic. The p-value is calculated using an empirical distribution of \( tTS \) obtained by simulating multivariate normal statistics with a covariance \( \Sigma \) as proposed by Jiang et al. (2011).

• TProd test - The procedure is similar to \( tTS \) with \( tProd \) defined as

\[
tProd = \prod p[i].
\]

See Zaykin et al. (2002) for details.

Value
A list with class "GGItest" containing the following components:

- statistic The value of the statistic: the p-value kept as the minimum after GATES correction
- p.value The p-value for the test
- estimate The estimation of the GATES p-value.
- alternative a character string describing the alternative.
- method a character string indicating the method used.
- data.name a character string giving the names of the data.

References

GBIGM.test


See Also

GGI

Examples

data(gene.pair)
## Estimation of the interaction between a pair of gene by using the Keff method with alpha=0.05.
gates.test(gene.pair$Y, gene.pair$G1, gene.pair$G2, me.est = "Keff",alpha=0.05)

---

GBIGM (Gene-based information gain method) for GGI analysis.

Description

GBIGM.test performs a Gene-Gene Interaction (GGI) analysis by contrasting the information entropy between cases and controls.

Usage

GBIGM.test(Y, G1, G2, n.perm = 1000)

Arguments

Y numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).

G1 SnpMatrix object. Must have a number of rows equal to the length of Y.

G2 SnpMatrix object. Must have a number of rows equal to the length of Y.

n.perm positive integer. n.perm is the number of permutations performed to compute the pvalue. By default, this is fixed to 1000.

Details

First, the conditional entropy and information gain rate are computed for each gene G1 and G2. In a second step, information gain rate for the gene pair (G1,G2) is computed. A p-value is estimated using permutations of Y. More details can be found in Li et al. (2015).

Value

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

- statistic The value of the statistic \( \Delta R_{1,2} \).
- p.value The p-value for the test.
- estimate The estimation of \( \Delta R_{1,2} \).
The number of permutations used to estimate the p-value.

 alternative

 a character string describing the alternative.

 method

 a character string indicating the method used.

 data.name

 a character string giving the names of the data.

References


See Also

GGI

Examples

data(gene.pair)
GGI.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,n.perm=500)

gene.pair A case-control dataset with two genes

Description

A case-control dataset containing the genotypes of 8 SNPs within GC gene (object G1) and 4 SNPs within PADI2 gene (object G2). The dataset includes 247 individuals affected by Rheumatoid Arthritis (RA) and 202 individuals not affected by RA.

Usage

data(gene.pair)

Format

A list with 3 objects:

Y A factor of disease status with two levels: HealthControl or RheumatoidArthritis.

G1 A SnpMatrix object with 8 SNPs.

G2 A SnpMatrix object with 4 SNPs.

Value

A dataset containing the genotypes of 8 SNPs within GC gene (object G1) and 4 SNPs within PADI2 gene (object G2).

Source

All three objects were taken from NCBI web site and are part of the GSE39428 series Chang X., et al. Investigating a pathogenic role for TXNDC5 in tumors. Int. J. Oncol., 43(6): 1871-84, 2013.
**GGI**

*Gene-Gene Interaction Analysis of a set of genes*

---

**Description**

GGI allows the search for Gene-Gene Interactions by testing all possible pairs of genes in a set of genes.

**Usage**

```r
GGI(Y, snpX, genes.length = NULL, genes.info = NULL, method = c("minP", "PCA", "CCA", "KCCA", "CLD", "PLSPM", "GBIGM", "GATES", "tTS", "tProd"), ...)
```

**Arguments**

- **Y**
  - numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of snpX (number of individuals).
- **snpX**
  - SnpMatrix object. Must have a number of rows equal to the length of Y. See details.
- **genes.length** *(optional)*
  - a numeric vector. gene.length is the length (in columns/SNP) of each gene.
- **genes.info** *(optional)*
  - a data frame. genes.info must have four columns named Genenames, SNPnames, Position and Chromosome. Each row describes a SNP and missing values are not allowed.
- **method**
  - a string matching one of the following: PCA, CCA, KCCA, CLD, PLSPM, GBIGM, minP, GATES, tTS or tProd. Only one method can be parsed.
- **...**
  - Other optional arguments to be passed to the functions associated with the method chosen. See more in elementary methods help.

**Details**

This function is a wrapper for all Gene-Gene Interaction analysis methods and drive the overall analysis: splitting the dataset in gene matrices and starting elementary analysis for each pair of genes.

*SNPs from the same gene are assumed to be ordered along the chromosome. See selectSnps.*

If genes.length is provided, it contains the number of SNPs of each gene. For example, if genes.length is the vector: c(20, 35, 15), then gene 1 will be interpreted as the set of the first 20 columns/SNPs of snpX, gene 2 will be interpreted as the following 35 columns/SNP, etc. Each gene declared is considered contiguous with the one before and after it. genes.length can be named if you want the returned matrix to have dimensions named after those. If no names are given then generic names are generated following the pattern Gene.n (n being the gene’s index).

The following methods are available to perform the interaction test for a single pair of genes:

- **Principal Components Analysis method (PCA)** PCA.test - PCA is performed on both genes and resulting principal components are used to fit a logistic regression model with interaction between and a second logistic regression model without interaction term. The interaction between the two genes is then tested using a likelihood ratio test between the two logistic regression models (see Li et al. 2009).
• Canonical Correlation Analysis (CCA) \texttt{CCA.test} - The maximum of canonical correlation between the two genes is computed for each group (cases and controls). The difference between the two transformed values (Fisher transformation) is used to test for interaction between genes (see Peng et al. 2010).

• Kernel Canonical Correlation Analysis (KCCA) \texttt{KCCA.test} - This method is similar to the CCA method where the canonical correlations are computed using Kernel method (see Yuan et al., 2012 and Larson et al., 2013).

• Composite Linkage Disequilibrium (CLD) \texttt{CLD.test} - CLD is based on the difference of the covariance matrices between the two genes computed for cases and controls. The covariance is estimated via the Composite Linkage Disequilibrium and a method based on Nagao normalized Quadratic Distance is used to compute the test statistic (see Rajapakse et al., 2012).

• Partial Least Square Path Modeling (PLSPM) \texttt{PLSPM.test} - A network of statistical relations between latent and manifest variables is built. The difference between the path coefficients is used to compute the test statistic (see Zhang et al., 2013).

• Gene-Based Information Gain Method (GBIGM) \texttt{GBIGM.test} - Entropies and Information Gain Ratio are used to compute a measure of the co-association between two genes (see Li et al., 2015).

• Minimum p-value test (minP) \texttt{minP.test} - Given two genes, \(G_1\) with \(m_1\) SNPs and \(G_2\) with \(m_2\) SNPs, all SNP-SNP interactions are first tested using a logistic regression model, thus generated a set of \(m_1 \times m_2\) p-values. The significance of the minimum p-value is evaluated using multivariate normal distribution that accounts for the covariance between the tests statistics at the SNP level (see Emily, 2016).

• Gene Association Test using Extended Simes procedure (GATES) \texttt{gates.test} - Given two genes, \(G_1\) with \(m_1\) SNPs and \(G_2\) with \(m_2\) SNPs, all SNP-SNP interactions are first tested using a logistic regression model, thus generated a set of \(m_1 \times m_2\) p-values. P-values are then corrected for multiple testing using an extension of the Simes procedure that take into account the correlation between the tests statistic via the number of effective tests (see Li. et al., 2011).

• Truncated Tail Strength test (tTS) \texttt{tTS.test} - Given two genes, \(G_1\) with \(m_1\) SNPs and \(G_2\) with \(m_2\) SNPs, all SNP-SNP interactions are first tested using a logistic regression model, thus generated a set of \(m_1 \times m_2\) p-values. All p-values below a user defined threshold are weighted and summed up to provide the tTS test statistic (see Jiang et al., 2011).

• Truncated p-value Product test (tProd) \texttt{tProd.test} - Similar to tTS but with a different p-values transformation (see Zaykin, 2002)

Missing values are not allowed and trying to parse an incomplete \texttt{SnpMatrix} object as an argument will result in an error. Imputation can be performed prior to the analysis with the \texttt{imputeSnpMatrix} function.

**Value**

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

- \texttt{statistic} - a symmetric matrix of size \(G \times G\) where \(G\) is the number of genes studied. The general term of the matrix is the statistic of the interaction between the two genes.

- \texttt{p.value} - a symmetric matrix of size \(G \times G\) where \(G\) is the number of genes studied. The general term of the matrix is the p-value of the interaction between the two genes.

- \texttt{df} - (Only for \texttt{method="PCA"}). a symmetric matrix of size \(G \times G\) where \(G\) is the number of genes studied. The general term of the matrix is the degrees of freedom of the interaction test.
method
The method used to perform the Gene-Gene interaction test.

parameter
A list of the parameters used to perform the Gene-Gene Interaction test.

References


See Also

PCA.test, CCA.test, KCCA.test, CLD.test, PLSPM.test, GBIGM.test, plot.GGInetwork, minP.test, gates.test, tTS.test, tProd.test, imputeSnpMatrix

Examples

## Not run:
## Dataset is included in the package
ped <- system.file("extdata/example.ped", package="GeneGeneInteR")
info <- system.file("extdata/example.info", package="GeneGeneInteR")
posi <- system.file("extdata/example.txt", package="GeneGeneInteR")

## Importation of the genotypes
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")
## Filtering of the data: SNPs with MAF < 0.05 or p.value for HWE < 1e-3 or SNPs with
## call.rate < 0.9 are removed.
data.scour <- snpMatrixScour(snpX=data.imported$snpX, genes.info=data.imported$genes.info,min.maf=0.05,
min.eq=1e-3,call.rate=0.9)
## Imputation of the missing genotypes
data.imputed <- imputeSnpMatrix(snpX=data.scour$snpX, genes.info = data.scour$genes.info)

## End(Not run)
## Equivalent loading of the genotypes
load(system.file("extdata/dataImputed.Rdata", package="GeneGeneInteR"))
### Importation of the phenotype

```r
resp <- system.file("extdata/response.txt", package="GeneGeneInteR")
Y <- read.csv(resp, header=FALSE)
```

### estimation of the interaction between the 17 genes with the CLD method -- can take a few minutes

```r
GGI.res <- GGI(Y=Y, snpX=data.imputed$snpX, genes.info=data.imputed$genes.info,method="CLD")
```

### estimation of the interaction between 12 among the 17 genes with the default PCA method

```r
dta <- selectSnps(data.imputed$snpX, data.imputed$genes.info, c("bub3","CDSN","Gc","GLRX", "PADI1","PADI2","PADI4","PADI6","PRKD3","PSORS1C1","SERPINA1","SORBS1"))
GGI.res <- GGI(Y=Y, snpX=dta$snpX, genes.info=dta$genes.info,method="PCA")
```

---

**importFile**

Imports SNPs information from pedfile, PLINK, VCF (4.0) file, or genotypes imputed by IMPUTE2.

---

**Description**

`importFile` generates a `SnpMatrix` object on the basis of diallelic object contained in a file, and creates also a data frame containing information about positions of the SNPs on the genome.

**Usage**

```r
importFile(file, pos, pos.sep = "\t", ...)
```

**Arguments**

- `file` String containing the path of the file to import.
- `pos` (optional) Path of a csv file, character vector or numeric vector containing informations for each SNP about chromosome, gene names, snp names and positions.
- `pos.sep` (optional) String to be passed to `read.csv` function as `sep` argument. Default is tab (\t).
- ... Additional arguments to be passed to the reading file function.

**Details**

As input information, `importFile` takes the full paths of the files to be imported. Files are then read with `read.pedfile`, `read.plink`, `vcf2sm`, or `read.impute`, depending on the file extension (pedfile, plink, vcf or impute2). For a pedfile, `importFile` also reads the ".info" file associated (this must be in the same directory as the ".ped". Similarly, for a PLINK, there must be 3 files with extensions ".bed" (passed as argument file), ".bim" and ".fam". A VCF file must be with the associated ".tbi" file.

If the file is a vcf file, two additional arguments have to done:

- `gr` instance of `GRanges`.

---

**Description**

As input information, `importFile` takes the full paths of the files to be imported. Files are then read with `read.pedfile`, `read.plink`, `vcf2sm`, or `read.impute`, depending on the file extension (pedfile, plink, vcf or impute2). For a pedfile, `importFile` also reads the ".info" file associated (this must be in the same directory as the ".ped". Similarly, for a PLINK, there must be 3 files with extensions ".bed" (passed as argument file), ".bim" and ".fam". A VCF file must be with the associated ".tbi" file.

If the file is a vcf file, two additional arguments have to done:

- `gr` instance of `GRanges`.

---

**Description**

As input information, `importFile` takes the full paths of the files to be imported. Files are then read with `read.pedfile`, `read.plink`, `vcf2sm`, or `read.impute`, depending on the file extension (pedfile, plink, vcf or impute2). For a pedfile, `importFile` also reads the ".info" file associated (this must be in the same directory as the ".ped". Similarly, for a PLINK, there must be 3 files with extensions ".bed" (passed as argument file), ".bim" and ".fam". A VCF file must be with the associated ".tbi" file.

If the file is a vcf file, two additional arguments have to done:

- `gr` instance of `GRanges`.

---

**Description**

As input information, `importFile` takes the full paths of the files to be imported. Files are then read with `read.pedfile`, `read.plink`, `vcf2sm`, or `read.impute`, depending on the file extension (pedfile, plink, vcf or impute2). For a pedfile, `importFile` also reads the ".info" file associated (this must be in the same directory as the ".ped". Similarly, for a PLINK, there must be 3 files with extensions ".bed" (passed as argument file), ".bim" and ".fam". A VCF file must be with the associated ".tbi" file.

If the file is a vcf file, two additional arguments have to done:

- `gr` instance of `GRanges`.
• nmetacol numeric defining the number of columns used in each record as locus-level metadata.

Pos argument is optional. If it’s not given, the data frame with position information is filled with NAs, except for SNP names which are imported from the column names of the SnpMatrix, and eventually positions and chromosomes if the file imported is a pedfile or a PLINK. Else, the pos argument can be either the path to a csv file, a character vector with elements of the form chr:position, or a numeric vector with only the positions. Additionnally, SNP names can be precised as names of the vector. If you choose the csv file path, be sure that the columns are named as follows: Chromosome, Genenames, SNPnames, Position.

Value

A list of two objects:

- snpX a SnpMatrix
- genes.info a data frame with 4 columns, and one row per SNP. The columns are Chromosome, Genenames, SNPnames and position.

Examples

```r
## Pedfile from this package.
ped <- system.file("extdata/example.ped", package="GeneGeneInteR")
info <- system.file("extdata/example.info", package="GeneGeneInteR")

## Information about position of the snps
posi <- system.file("extdata/example.txt", package="GeneGeneInteR")

## Importation
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")

##########
## VCF file from GGtools package.
## Not run:
vref <- system.file("vcf/CEU.exon.2010_09.genotypes.vcf.gz", package="GGtools")
irange <- IRanges::IRanges(10e6,20e6)

## Not run:
gg = GenomicRanges::GRanges(seqnames="1", ranges=irange)

da <- importFile(file=vref, gr=gg, nmetacol=9L)
## End(Not run)
```

**imputeSnpMatrix**

Missing values handling for SnpMatrix object

Description

imputeSnpMatrix is a generic wrapper of snp.imputation and impute.snps functions from snpStats package. This function mimics a Leave-One-Out process where missing SNPs are imputed for an individual based on a model trained on all other individuals.

Usage

```r
imputeSnpMatrix(snpX, genes.info, on.rem = c("SNP", "ind", "none"), quiet = FALSE)
```
Arguments

snpX  snpMatrix object of which SNPs are to be removed
genes.info  A data.frame with four columns named Genenames, SNPnames, Position and Chromosome. Each row describes a SNP and missing values are not allowed.
on.rem  (optional) a character string matching one of the following items: SNP, ind, none. Describes the action taken in case of remaining missing values. See details
quiet  (optional) a boolean describing whether or not progress bar should be displayed.

Details

For the ith row in the snpX argument (i.e. the ith individual in the dataset), the following steps are performed:

- missing SNPs are detected for individual i
- rules are imputed for every missing SNPs using the whole dataset where individual i is removed
- SNPs are imputed for individual i

Although, such a process allows the imputation of a large number of missing SNPs, some missing values may remain. In that case, the user can specify the action to be done thanks to the om.rem arguments:

- om.rem="none": leave the dataset as it is,
- om.rem="SNP": remove all SNPs with remaining missing values,
- om.rem="ind": remove all individuals with remaining missing values.

Removing all SNPs is often more parsimonious than removing individuals and allows to get a dataset without any missing values with minimum information-loss.

Value

A list object with two named elements:

snpX  a SnpMatrix object corresponding to input matrix with imputed values.
genes.info  a data frame object corresponding to an updated version of input genes.info in case SNP had to be removed.

A warning is printed when SNPs or individuals are removed.

See Also

GGI snpMatrixScour

Examples

## Not run:
ped <- system.file("extdata/example.ped", package="GeneGeneInteR")
info <- system.file("extdata/example.info", package="GeneGeneInteR")
posi <- system.file("extdata/example.txt", package="GeneGeneInteR")
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")

## Example 1 without SNPs filtering
## In this example, 9 SNPs were removed due to remaining missing values.
KCCA.test

KCCA.test <- imputeSnpMatrix(data.imported$snpX, genes.info = data.imported$genes.info)

## Example 2 with SNPs filtering priori to the imputation
## Filtering of the data: SNPs with MAF < 0.05 or p.value for HWE < 1e-3 or SNPs with
call.rate < 0.9 are removed.
data.scour <- snpMatrixScour(snpX=dta$snpX,genes.info=dta$genes.info,min.maf=0.05,
                          min.eq=1e-3,call.rate=0)

## End(Not run)

## Equivalent loading of data for example 2
## Imputation of the missing genotypes
load(system.file("extdata/dataScour.Rdata", package="GeneGeneInteR"))
data.imputed <- imputeSnpMatrix(data.scour$snpX, genes.info = data.scour$genes.info)

---

KCCA.test

KCCA (Kernel Canonical-Correlation Analysis) based GGI analysis.

Description

KCCA.test performs a Gene-Gene Interaction (GGI) analysis based on the difference of canonical
correlations between cases and controls. The "kernel trick" is applied to the canonical correlation
to allow the detection of non-linear co-association.

Usage

KCCA.test(Y, G1, G2,kernel=c("rbfdot","polydot","tanhdot","vanilladot","laplacedot",
                          "besseldot","anovadot","splinedot"),n.boot = 500,sigma=0.05,degree=1,scale=1,offset=1,
           order=1)

Arguments

Y numeric or factor vector with exactly two different values. Y is the response
variable and should be of length equal to the number of rows of G1 and G2
arguments (number of individuals).

G1 SnpMatrix object. Must have a number of rows equal to the length of Y.

G2 SnpMatrix object. Must have a number of rows equal to the length of Y.

kernel A character string matching one of the kernel name in : "rbfdot","polydot","tanhdot","vanilladot","laplacedot",
         "besseldot","anovadot","splinedot". For more details regarding kernel function see dots object. Default is a Gaussian
         RBF kernel

n.boot positive integer. n.boot is the number of bootstrap replicates for estimating
variances. By default, this is fixed to 500.

sigma The inverse kernel width used by the Gaussian the Laplacian (kernel="rbfdot"),
         the Bessel (kernel="besseldot") and the ANOVA (kernel="anovadot") kernel.
         Default is sigma=0.05.

degree The degree of the polynomial (kernel="polydot"), bessel (kernel="besseldot")
or ANOVA (kernel="anovadot") kernel function. This has to be an positive
integer. Default is degree=1.
scale  The scaling parameter of the polynomial (kernel="polydot") and tangent (kernel="tanhdot")
kernel is a convenient way of normalizing patterns without the need to modify
the data itself. Default is scale=1.

offset  The offset used in a polynomial (kernel="polydot") or hyperbolic tangent
(kernel="tanhdot") kernel. Default is offset=1.

order  The order of the Bessel function to be used as a kernel (kernel="besseldot").
Default is order=1.

Details
The test statistic is based on the difference between a Fisher’s transformation of the maximum of
the kernelized canonical correlations in cases and controls. To calculate the test statistic for the
interaction pvalue, KCCA.test estimates the variance of the Fisher’s transformation of the maximum
of the kernelized canonical correlations in cases and controls using a bootstrap method. The
computation of kcca can be very long.

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

statistic  The value of the statistic KCCU.
p.value  The p-value for the test.
estimate  A vector of the Fisher’s transformed maximum kernel canonical correlation co-
efficient in cases and controls.
parameter  The number of bootstrap samples used to estimate the p-value.
null.value  The value of KCCU under the null hypothesis.
alternative  a character string describing the alternative.
method  a character string indicating the method used.
data.name  a character string giving the names of the data.

References
Yuan, Z. et al. (2012): Detection for gene-gene co-association via kernel canonical correlation anal-
ysis, BMC Genetics, 13, 83.
Larson, N. B. et al. (2013): A kernel regression approach to gene-gene interaction detection for
case-control studies, Genetic Epidemiology, 37, 695-703.

See Also
GGI, CCA.test

Examples
data(gene.pair)
## Not run:
KCCA.test(Y=gene.pair$Y, G1=gene.pair$G1, G2=gene.pair$G2)
## End(Not run)
**minP.test**

**Gene-based Gene-Gene Interaction analysis by combining SNP-SNP interaction tests**

**Description**

gates.test, minP.test, tTS.test and tProd.test aim at performing gene-gene interaction analysis based on SNP-SNP interaction tests. The following methods are used to combine SNP-SNP interaction tests into a single Gene-Gene Interaction p-value:

- Minimum p-value in minP.test function
- Gene Association Test with Extended Simes procedure in gates.test
- Truncated Tail Strength in tTS.test function
- Truncated p-value Product in tProd.test function

**Usage**

```r
minP.test(Y, G1, G2)
```

**Arguments**

- `Y` numeric or factor vector with exactly two different values. `Y` is the response variable and should be of length equal to the number of rows of `G1` and `G2` arguments (number of individuals).
- `G1` SnpMatrix object. Must have a number of rows equal to the length of `Y`.
- `G2` SnpMatrix object. Must have a number of rows equal to the length of `Y`.

**Details**

In a first step, all methods start by applying a logistic regression model to test all pairs of SNPs between the two genes `G1` and `G2`. If `G1` has `m1` SNPs and `G2` `m2` SNPs, a total of `m1 * m2` SNP-SNP tests are performed. In a second step, the `m1 * m2` SNP-SNP tests are combined according to their covariance matrix `Σ`. `Σ` is computed as described in the method developed in Emily (2016). The covariance `Σ` is used in each method as follows:

- **minP test** - minP test considered the significant of the observed minimum p-value. Significance is computed by integrating the multivariate normal distribution with covariance `Σ` as proposed in Conneelly and Boehnke (2008).
- **GATES test** - The p-value for GATES is the minimum p-value obtained after a multiple testing correction of the SNP-SNP interaction p-values. Correction for multiple testing is defined as

\[
me * p[i]/me[i]
\]

where `me` is the effective number of independent tests, `p[i]` is the i-th top p-values and `me[i]` is the effective number of independent test among the top i p-values. Many methods exist to estimate `me` and `me[i]` terms:

- Cheverud-Nyholt method (Cheverud, 2001 and Nyholt, 2004)
- Keff method (Moskovina and Schmidt, 2008)
- Li & Ji method (Li and Ji, 2005)
- Galwey method (Galwey, 2009)
Details of each method can be found in the references.

- **tTS test** - tTS test does not consider only the strongest signal but all signals that are inferior to a given threshold $\tau$. For these p-values, the weighted sum of

$$tTS = \sum (1 - p[i] \ast (m1 \ast m2 + 1)/i)$$

is computed and represents the test statistic. The p-value is calculated using an empirical distribution of $tTS$ obtained by simulating multivariate normal statistics with a covariance $\Sigma$ as proposed by Jiang et al. (2011).

- **TProd test** - The procedure is similar to $tTS$ with $tProd$ defined as

$$tProd = \prod p[i]$$

See Zaykin et al. (2002) for details.

**Value**

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

- **statistic**  The value of the statistic: the maximum of the absolute value of the observed pairwised statistics.
- **p.value**  The p-value for the test.
- **estimate**  The estimation of the maximum of the absolute value of the observed pairwised statistics.
- **alternative**  a character string describing the alternative.
- **method**  a character string indicating the method used.
- **data.name**  a character string giving the names of the data.

**References**


See Also
GGI

Examples

```r
data(gene.pair)
minP.test(gene.pair$Y, gene.pair$G1, gene.pair$G2)
```

PCA.test
PCA-based Gene-Gene Interaction analysis.

Description

PCA.test performs a Gene-Gene Interaction (GGI) analysis by testing the interaction between the principal components of the two genes. With method="Std" PCA is standardized using standard deviation for each variable. With method="GenFreq", dataset is standardized using the standard deviation under Hardy-Weinberg equilibrium, as proposed in the snpStats Bioconductor package.

Usage

```r
PCA.test(Y, G1, G2, threshold = 0.8, method="GenFreq")
```

Arguments

- `Y` numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).
- `G1` SnpMatrix object. Must have a number of rows equal to the length of Y.
- `G2` SnpMatrix object. Must have a number of rows equal to the length of Y.
- `threshold` (optional) numeric comprised in [0,1] interval. threshold is the minimum percentage of inertia kept for each PCA.
- `method` (optional) character string for PCA method. Must be one of the following: "GenFreq", "Std" (See details).

Details

In a first step, PCA.test performs a Principal Components Analysis on both G1 and G2 genes that are interpreted as matrices of allele counts. With the method="Std", the dataset is standardized using variables standard deviation, while dataset is standardized using standard deviation under Hardy-Weinberg equilibrium for method="GenFreq". Principal components are then retrieved to describe each dataset with user-defined inertia percentage (parameter threshold) and used in a logistic regression model. The consists is testing the significance of the interaction terms using a Likelihood Ratio Test (see Li et al. (2009)).
plot.GGInetwork

Value

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

- statistic: The value of the statistic: the deviance of the anova test.
- p.value: The p-value for the test.
- estimate: A vector of the residual deviances.
- parameter: The degrees of freedom of the chi-squared distribution of the test statistic.
- null.value: The value of the deviance under the null.
- alternative: A character string describing the alternative.
- method: A character string indicating the method used.
- data.name: A character string giving the names of the data.

References


See Also

GGI

Examples

data(gene.pair)
PCA.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,threshold=0.7,method="Std")
PCA.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,threshold=0.7,method="GenFreq")

plot.GGInetwork

Plot a Gene-Gene Interaction Study with an heatmap-like visualization

Description

GGI.plot is a graphical function that allow a heatmap-like or a network-like visualization of a Gene-Gene Interaction study based on a set of genes.

Usage

```r
## S3 method for class 'GGInetwork'
plot(x, method=c("heatmap", "network"), threshold=NULL, col=c("#D6604D", "#104E8B"), colbar.width=0.15, title=NULL, hclust.order=FALSE, use.log=FALSE, NA.col="#D3D3D3", draw.pvals=NULL, draw.names=NULL, interact=FALSE, method.adjust=c("none", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr"), genes=1:ncol(x$p.value), plot.nointer=TRUE, ...)
```
Arguments

x  GGInetwork. Given as output of GGI.

method  Output graph ("heatmap" for heatmap-like, "network" for network-like). Default is heatmap.

threshold  A numeric between 0 and 1. All p-value strictly greater than that threshold are distinctly colored (See NA.col). Default is NULL corresponding to no threshold applied for an heatmap graph and a 0.05 threshold applied for the network representation.

method.adjust  correction method for multiple testing as proposed in the p.adjust function. By default, no correction is applied to the p-values.

genes  Numeric vector allowing a selection of the genes that will be included in the relations. Default is set to all genes.

col  (Only for method=heatmap.) String vector. Marker colors to be used for the gradient. The first element of the vector is the value for 0 and the last is for 1. If only one value is parsed, that color is used for 0 and white is automatically used for 1. Any value compatible with colorRampPalette function.

colbar.width  (Only for method=heatmap.) A positive number describing the gradient bar width ratio. That number is used to keep the gradient bar’s width steady as the size of GGI increases.

title  (Only for method=heatmap.) A string used as the plot title. If left as NULL, a generic name is generated.

hclust.order  (Only for method=heatmap.) A boolean. Should a hierachical clustering procedure be performed on GGI to order the matrix ?

use.log  (Only for method=heatmap.) A boolean. Should the p-values be reported in -log10? FALSE by default.

NA.col  (Only for method=heatmap.) A string. The color used when a p-value is strictly greater than threshold.

draw.pvals  (Only for method=heatmap.) Either NULL or a boolean. Should p-values be plotted ? Disabled when the size of GGI exceeds 15 * 15.

draw.names  (Only for method=heatmap.) Either NULL or a boolean. Should genes’ names be plotted on matrix margins? Disabled when the size of GGI exceeds 25 * 25.

interact  (Only for method=heatmap.) A boolean. Should the plot be clickable ? (See Details for more information). Disabled when open R session is not interactive.

plot.nointer  (Only for method=network.) A boolean. Set TRUE if the genes with no interaction should be plotted.

...  further arguments passed to or from other methods.

Details

If method=heatmap, this function draw the upper half of a Gene-Gene Interaction results matrix without its diagonal. A gradient is created from 0 to 1 (by default from crimson to white) and the matrix cells are colored according to the corresponding p-value.

By default, when draw.pvals=NULL and draw.names=NULL, p-values and names are drawn to make matrix reading easier, but in case parameter GGI is large, p-values (and eventually gene names as GGI grows bigger) are not drawn. In that case, the default behavior of the function is to start an interactive process where user can click on a cell of interest to open a tooltip displaying which genes are involved in selected interaction and the p-value of the interaction test. Tooltips can be closed if
user clicks anywhere else than on a cell. This process stops when the user presses the escape button (or terminates the locator procedure in general) or when the user clicks on any place other than a cell when no tooltip window is open.

To improve plot clarity, user may set a threshold above which cells are colored with a distinct color. By default, threshold is set to 1 and no cell is colored differently (as values must be strictly above the threshold).

If method=network, this function plots a graph representing the significative interactions between genes of a Gene-Gene Interaction study.

Value

The form of the value returned by plot depends on the class of its argument. See Details.

See Also

GGI

Examples

## Not run:
## Dataset is included in the package
ped <- system.file("extdata/example.ped", package="GeneGeneInteR")
info <- system.file("extdata/example.info", package="GeneGeneInteR")
posi <- system.file("extdata/example.txt", package="GeneGeneInteR")

## Importation of the genotypes
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")
## Filtering of the data: SNPs with MAF < 0.05 or p.value for HWE < 1e-3 or
call rate < 0.9 are removed.
data.scour <- snpMatrixScour(snpX=data.imported$snpX,genes.info=data.imported$genes.info, min.maf=0.05,min.eq=1e-3,call.rate=0.9)
## Imputation of the missing genotypes
data.imputed <- imputeSnpMatrix(data.scour$snpX, genes.info = data.scour$genes.info)

## Importation of the phenotype
resp <- system.file("extdata/response.txt", package="GeneGeneInteR")
Y <- read.csv(resp, header=FALSE)

## plot of the interaction between the 17 genes with the CLD method -- can take a few minutes
GGI.res <- GGI(Y=Y, snpX=data.imputed$snpX, genes.info=data.imputed$genes.info,method="CLD")
plot(GGI.res,threshold=0.05)

## Selection of 12 genes among 17
data.select <- selectSnps(data.imputed$snpX, data.imputed$genes.info, c("bub3","CDSN","Gc","GLRX", "PADI1","PADI2","PADI4","PADI6","PRKD3","PSORS1C1","SERPINA1","SORBS1"))
GGI.res <- GGI(Y=Y, snpX=data.select$snpX, genes.info=data.select$genes.info,method="PCA")

## End(Not run)
## Equivalent importation of the GGI.res object
load(system.file("extdata/GGIRes.Rdata", package="GeneGeneInteR"))

## Plot of the results with default values
plot(GGI.res)

## Plot of the results with a threshold and an ordering of the genes.
## Default method is an heatmap-like representation
plot(GGI.res,threshold=0.1,hclust.order=TRUE)

## Example of network with default threshold 0.05
plot(GGI.res,method="network")

## Example of network with threshold 0.01 where genes with no interaction are not plotted
# (plot.nointer=FALSE)
plot(GGI.res,method="network",threshold=0.1,plot.nointer=FALSE)

---

**PLSPM.test**

**PLSPM (Partial Least Squares Path Modelling Analysis) based Gene-Gene Interaction analysis.**

### Description

PLSPM.test performs a Gene-Gene Interaction (GGI) analysis based on the modelisation of a network of statistical relations. The aim is to quantify the connections between the latent and the manifest variables.

### Usage

PLSPM.test(Y, G1, G2, n.perm=500)

### Arguments

- **Y**: numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).
- **G1**: SnpMatrix object. Must have a number of rows equal to the length of Y.
- **G2**: SnpMatrix object. Must have a number of rows equal to the length of Y.
- **n.perm**: positive integer. n.perm is the number of permutations performed to compute the pvalue. By default, this is fixed to 500.

### Details

The PLSPM-based method, as described in Zhang et al. (2013), aims at comparing the path coefficients $\beta_D$ and $\beta_C$, where $\beta_D$ is calculated among cases and $\beta_C$ among controls. The co-association between genes G1 and G2 is defined by:

$$U = \frac{\beta_D - \beta_C}{\sqrt{Var(\beta_D - \beta_C)}}$$

The plspm R package is used to estimate $U$. The significance pvalue is obtained by using a permutation method on the difference between the path coefficients.
Value

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

- **statistic**: The value of the statistic U.
- **p.value**: The p-value for the test.
- **estimate**: A vector of the path coefficients in cases and controls.
- **parameter**: The number of bootstrap samples used to estimate the p-value.
- **null.value**: The value of U under the null hypothesis.
- **alternative**: A character string describing the alternative.
- **method**: A character string indicating the method used.
- **data.name**: A character string giving the names of the data.

References


See Also

GGI

Examples

data(gene.pair)
PLSPM.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,n.perm=50)

print.GGItest

Printing objects of class "GGItest"

Description

Printing objects of class "GGItest"

Usage

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

Arguments

- **x**: GGItest. Given as output of `CCA.test` for example.
- **...**: further arguments passed to or from other methods

Details

`print.GGItest` provides a customized output of a GGItest object.
selectSnps

Value

the argument x, invisibly, as for all print methods.

Examples

data(gene.pair)
print(PCA.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,threshold=0.7,method="Std"))

selectSnps  Selects SNPs in snpMatrix and associated info data.frame, outputs of importFile.

Description

selectSnps allows the user to select snps from an object output of importFile. It generates the same object, with the columns of the snpMatrix and the rows of the data.frame corresponding to the selected snps.

Usage

selectSnps(snpX, genes.info, select)

Arguments

snpX  snpMatrix object. Given as output of importFile.

genes.info  Data.frame containing informations about snps. For more details, refer to importFile help file.

select  Numeric or character vector for selecting snps in snpX and genes.info. See details for more information.

Details

The column names of the genes.info data.frame should correspond to the output genes.info object returned by importFile function.

The select argument should one of the following:

• a numeric vector with only the column number in the snpMatrix (or row number for genes.info) of each snp selected.
• a character vector with the names of each snp selected or each gene selected.
• a character vector which elements are position bounds of genes. Each element of the vector is either of the form "begin:end", or "chr:begin:end" if you have to precise the chromosome of the gene.
snpMatrixScour

Value

A list of two objects:

- `snpX` a snpMatrix
- `genes.info` a data frame with 4 columns, and one row per SNP selected with `select`. The columns are Chromosome, Genenames, SNPnames and Position.

An error message is displayed if the genes of snps selected are not found in the either `snpX` or `genes.info`.

Examples

```r
## Importation of the dataset
## Not run:
ped <- system.file("extdata/example.ped", package="GeneGeneInteR")
info <- system.file("extdata/example.info", package="GeneGeneInteR")
posi <- system.file("extdata/example.txt", package="GeneGeneInteR")
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")
## End(Not run)

## Equivalent loading of the imported data
load(system.file("extdata/dataImported.Rdata", package="GeneGeneInteR"))

## Selection of the genes DNAH9 and TXNDC5
selec <- selectSnps(data.imported$snpX, data.imported$genes.info, c("DNAH9","TXNDC5"))

## Selection of the snps from position 101342000 to 101490000 on chromosome 15
selec <- selectSnps(data.imported$snpX, data.imported$genes.info, c("15:101342000:101490000"))
```

---

snpMatrixScour  
**SNP filtering based on Minor Allele Frequency, Hardy-Weinberg Equilibrium and call rate**

Description

snpMatrixScour aims at filtering out SNPs of a snpMatrix object based on Minor Allele Frequency criterion, deviation to Hardy-Weinberg Equilibrium and SNPs call rate.

Usage

```r
snpMatrixScour(snpX, genes.length = NULL, genes.info = NULL, min.maf = 0.01, min.eq = 0.01, call.rate = 0.9)
```

Arguments

- `snpX`  
  SnpMatrix object from which SNPs are to be removed
- `genes.length`  
  (optional) numeric vector. It is the length (in columns/SNPs) of each gene. Each gene declared is considered contiguous with the one before and after it. genes.lengths can be named (names will be kept).
- `genes.info`  
  (optional) a data.frame with four columns named Genenames, SNPnames, Position and Chromosome. Each row describes a SNP and missing values are not allowed.
**snpMatrixScour**

- **min.maf** a single value between 0 and 0.5 that gives the threshold for the MAF (Minor Allele Frequency) of a SNP. SNP with MAF < min.maf are removed. Default is 0.01.
- **min.eq** a single value between 0 and 1 that gives the maximum acceptable p-value for the \(\chi^2\) verifying HWE deviation. SNP that does not meet that criterion are removed. Default is 0.01.
- **call.rate** a single value between 0 and 1 that gives the minimum acceptable call rate for a SNP. Default is 0.9. Low values for SNPs call rate can make imputation harder (residual missing values).

**Details**

This function removes SNPs that does not meet all following criteria:

- \(MAF < \text{min.maf}\),
- \(HWE < \text{min.eq}\), where HWE is the p-value of the \(\chi^2\) test of deviation to Hardy-Weinberg Equilibrium,
- Call rate > call.rate.

If genes.length and genes.info are provided by the user, an updated version is returned by snpMatrixScour. The returned object can be directly used as inputs of the GGI function.

**Value**

A list with two objects:

- **snpX** the SnpMatrix object where non-conform SNPs are removed.
- **genes.info** the object that contains the updated gene lengths information. Can be a numeric vector (possibly named) or a data frame. If genes.length and genes.info are not provided by the user as input of the snpMatrixScour function, the genes.info object is NULL.

**See Also**

GGI

**Examples**

```r
## Not run:
ped <- system.file("extdata/example.ped", package="GeneGeneInteR")
info <- system.file("extdata/example.info", package="GeneGeneInteR")
posi <- system.file("extdata/example.txt", package="GeneGeneInteR")
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")

## End(Not run)
### Equivalent loading of the imported data
load(system.file("extdata/dataImported.Rdata", package="GeneGeneInteR"))

## In this example, SNPs are with MAF lower than 0.2 or p-value for HWE testing lower than 0.05 or # a proportion of missing value higher than 0.2 are removed
data.scour1 <- snpMatrixScour(data.imported$snpX, genes.info = data.imported$genes.info, min.maf = 0.2, min.eq=0.05, call.rate = 0.8)
## Two genes have been completely removed from the resulting dataset.
```
summary.GGInetwork

Printing summaries class "GGInetwork" objects

Description

Printing summaries for objects of class "GGInetwork"

Usage

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

Arguments

- `object`: GGInetwork. Given as output of `GGI`.
- `...`: further arguments passed to or from other methods

Details

`summary.GGI` provides a customized summary of a `GGInetwork` object.

Value

The form of the value returned by `summary` depends on the class of its argument.

Examples

```r
## Not run:
## Dataset is included in the package
data.imported <- importFile(file=ped, snps=info, pos=posi, pos.sep="\t")
data.scour <- snpMatrixScour(snpX=data.imported$snpX, genes.info=data.imported$genes.info,
                           min.maf=0.05, min.eq=1e-3, call.rate=0.9)
data.imputed <- imputeSnpMatrix(data.scour$snpX, genes.info = data.scour$genes.info)

## Importation of the phenotype
resp <- system.file("extdata/response.txt", package="GeneGeneInteR")
Y <- read.csv(resp, header=FALSE)

## plot of the interaction between the 17 genes with the CLD method -- can take a few minutes
GGI.res <- GGI(Y=Y, snpX=dt$a$snpX, genes.info=dt$a$genes.info, method="CLD")
plot(GGI.res, threshold=0.05)

## Selection of 12 genes among 17
data.select <- selectSnps(data.imputed$snpX, data.imputed$genes.info, c("bub3", "CDSN", "Gc", "GLRX",
   "PADI1", "PADI2", "PADI4", "PADI6", "PRKD3", "PSORS1C1", "SERPINA1", "SORBS1"))
```
summary.GGItest

GGI.res <- GGI(Y=Y, snpX=data.select$snpX, genes.info=data.select$genes.info, method="PCA")

## End(Not run)
## Equivalent importation of the GGI.res object
load(system.file("extdata/GGIRes.Rdata", package="GeneGeneInteR"))

summary(GGI.res)

---

**summary.GGItest**  
**Printing summaries class "GGItest" objects**

**Description**

Printing summaries for objects of class "GGItest"

**Usage**

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

**Arguments**

- `object`  
  GGItest. Given as output of `CCA.test` for example.

- `...`  
  further arguments passed to or from other methods

**Details**

`summary.GGItest` provides a customized summary of a GGItest object.

**Value**

The form of the value returned by `summary` depends on the class of its argument.

**Examples**

```r
data(gene.pair)
summary(PCA.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2, threshold=0.7, method="Std"))
```
tProd.test

Gene-based Gene-Gene Interaction analysis by combining SNP-SNP interaction tests

Description

gates.test, minP.test, tTS.test and tProd.test aim at performing gene-gene interaction analysis based on SNP-SNP interaction tests. The following methods are used to combine SNP-SNP interaction tests into a single Gene-Gene Interaction p-value:

- Minimum p-value in minP.test function
- Gene Association Test with Extended Simes procedure in gates.test
- Truncated Tail Strength in tTS.test function
- Truncated p-value Product in tProd.test function

Usage

tProd.test(Y, G1, G2, tau = 0.05, n.sim = 1000)

Arguments

- **Y**: numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).
- **G1**: SnpMatrix object. Must have a number of rows equal to the length of Y.
- **G2**: SnpMatrix object. Must have a number of rows equal to the length of Y.
- **tau**: numeric in [0, 1]. See details section for its use.
- **n.sim**: positive integer. n.sim is the number of multivariate normal distribution simulations used to compute the significant level for tTS and tProd methods.

Details

In a first step, all methods start by applying a logistic regression model to test all pairs of SNPs between the two genes G1 and G2. If G1 has \(m1\) SNPs and G2 \(m2\) SNPs, a total of \(m1 \times m2\) SNP-SNP tests are performed. In a second step, the \(m1 \times m2\) SNP-SNP tests are combined according to their covariance matrix \(\Sigma\). \(\Sigma\) is computed as described in the method developed in Emily (2016). The covariance \(\Sigma\) is used in each method as follows:

- **minP test** - minP test considered the significant of the observed minimum p-value. Significance is computed by integrating the multivariate normal distribution with covariance \(\Sigma\) as proposed in Conneely and Boehnke (2008).
- **GATES test** - The p-value for GATES is the minimum p-value obtained after a multiple testing correction of the SNP-SNP interaction p-values. Correction for multiple testing is defined as

\[
me \times p[i]/me[i]
\]

where \(me\) is the effective number of independent tests, \(p[i]\) is the i-th top p-values and \(me[i]\) is the effective number of independent test among the top i p-values. Many methods exist to estimate \(me\) and \(me[i]\) terms:
tProd.test

- Cheverud-Nyholt method (Cheverud, 2001 and Nyholt, 2004)
- Keff method (Moskovina and Schmidt, 2008)
- Li & Ji method (Li and Ji, 2005)
- Galwey method (Galwey, 2009)

Details of each method can be found in the references.

• tTS test - tTS test does not consider only the strongest signal but all signals that are inferior to a given threshold $\tau$. For these $p$-values, the weighted sum of

$$tTS = \sum (1 - p[i] \ast (m1 + m2 + 1)/i)$$

is computed and represents the test statistic. The $p$-value is calculated using an empirical distribution of $tTS$ obtained by simulating multivariate normal statistics with a covariance $\Sigma$ as proposed by Jiang et al. (2011).

• TProd test - The procedure is similar to $tTS$ with $tProd$ defined as

$$tProd = \prod p[i].$$

See Zaykin et al. (2002) for details.

Value

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

- statistic: The value of the statistic tProd.
- p.value: The p-value for the test.
- estimate: Estimation of tProd.
- parameter: The threshold value tau.
- alternative: a character string describing the alternative.
- method: a character string indicating the method used.
- data.name: a character string giving the names of the data.

References

K.N. Conneely and M. Boehnke. So many correlated tests, so little time! rapid adjustment of p values for multiple correlated tests. The American Journal of Human Genetics, 81: 1158-1168.
See Also

GGI

Examples

data(gene.pair)
tProd.test(gene.pair$Y, gene.pair$G1, gene.pair$G2, tau = 0.5, n.sim = 500)

tTS.test

Gene-based Gene-Gene Interaction analysis by combining SNP-SNP interaction tests

Description

gates.test, minP.test, tTS.test and tProd.test aim at performing gene-gene interaction analysis based on SNP-SNP interaction tests. The following methods are used to combine SNP-SNP interaction tests into a single Gene-Gene Interaction p-value:

• Minimum p-value in minP.test function
• Gene Association Test with Extended Simes procedure in gates.test
• Truncated Tail Strength in tTS.test function
• Truncated p-value Product in tProd.test function

Usage

tTS.test(Y, G1, G2, tau = 0.05, n.sim = 1000)

Arguments

Y  numeric or factor vector with exactly two different values. Y is the response variable and should be of length equal to the number of rows of G1 and G2 arguments (number of individuals).

G1  SnpMatrix object. Must have a number of rows equal to the length of Y.

G2  SnpMatrix object. Must have a number of rows equal to the length of Y.

tau  numeric in [0, 1]. See details section for its use.

n.sim  positive integer. n.sim is the number of multivariate normal distribution simulations used to compute the significant level for tTS and tProd methods.
Details

In a first step, all methods start by applying a logistic regression model to test all pairs of SNPs between the two genes $G_1$ and $G_2$. If $G_1$ has $m_1$ SNPs and $G_2$ $m_2$ SNPs, a total of $m_1 \times m_2$ SNP-SNP tests are performed. In a second step, the $m_1 \times m_2$ SNP-SNP tests are combined according to their covariance matrix $\Sigma$. $\Sigma$ is computed as described in the method developed in Emily (2016).

The covariance $\Sigma$ is used in each method as follows:

- **minP test** - minP test considered the significant of the observed minimum p-value. Significance is computed by integrating the multivariate normal distribution with covariance $\Sigma$ as proposed in Conneely and Boehnke (2008).

- **GATES test** - The p-value for GATES is the minimum p-value obtained after a multiple testing correction of the SNP-SNP interaction p-values. Correction for multiple testing is defined as

  $$ me \times p[i] / me[i] $$

  where $me$ is the effective number of independant tests, $p[i]$ is the i-th top p-values and $me[i]$ is the effective number of independant test among the top i p-values. Many methods exist to estimate $me$ and $me[i]$ terms:

  - Cheverud-Nyholt method (Cheverud, 2001 and Nyholt, 2004)
  - Keff method (Moskovina and Schmidt, 2008)
  - Li & Ji method (Li and Ji, 2005)
  - Galwey method (Galwey, 2009)

  Details of each method can be found in the references.

- **tTS test** - tTS test does not consider only the strongest signal but all signals that are inferior to a given threshold $\tau$. For these p-values, the weighted sum of

  $$ tTS = \sum \left(1 - p[i] \times (m_1 \times m_2 + 1) / i \right) $$

  is computed and represents the test statistic. The p-value is calculated using an empirical distribution of $tTS$ obtained by simulating multivariate normal statistics with a covariance $\Sigma$ as proposed by Jiang et al. (2011).

- **TProd test** - The procedure is similar to $tTS$ with $tProd$ defined as

  $$ tProd = \prod p[i]. $$

  See Zaykin et al. (2002) for details.

Value

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

- **statistic** - The value of the statistic tTS.
- **p.value** - The p-value for the test.
- **estimate** - Estimation of tTS.
- **parameter** - The threshold value tau.
- **alternative** - a character string describing the alternative.
- **method** - a character string indicating the method used.
- **data.name** - a character string giving the names of the data.
References


See Also

GGI

Examples

data(gene.pair)
tTS.test(gene.pair$Y, gene.pair$G1, gene.pair$G2, tau = 0.5, n.sim = 500)
Index

*Topic datasets
    data.SNP, 4
    gene.pair, 8
CCA.test, 2, 10, 11, 16, 24, 29
CLD.test, 3, 10, 11
colorRampPalette, 21
data.frame, 14, 26
data.SNP, 4
dots, 15
factor, 8
gates.test, 5, 10, 11
GBIGM.test, 7, 10, 11
gene.pair, 8
GGI, 3, 4, 7, 8, 9, 14, 16, 19–22, 24, 27, 28, 32, 34
GRanges, 12
importFile, 12, 25
imputeSnpMatrix, 10, 11, 13
kcca, 16
KCCA.test, 3, 10, 11, 15
matrix, 10
minP.test, 10, 11, 17
p.adjust, 21
PCA.test, 9, 11, 19
plot.GGInetwork, 11, 20
plspm, 23
PLSPM.test, 10, 11, 23
print, 25
print.GGItest, 24
read.impute, 12
read.pedfile, 12
read.plink, 12
selectSnps, 9, 25
snpMatrixScour, 14, 26
summary.GGItest, 29
tProd.test, 10, 11, 30
tTSS.test, 10, 11, 32
vcf2sm, 12
summary.GGInetwork, 28