Package ‘GeneScan3DKnock’

November 13, 2020

Type Package

Title A Unified Framework for Gene-Based Testing with Joint Analysis of Coding and Regulatory Variation, and Integration of Knockoff Statistics for Causal Gene Identification

Version 0.1

Author Shiyang Ma, Zihuai He, James Dalgleish

Maintainer Shiyang Ma <sm4857@cumc.columbia.edu>

Description Functions for the gene-based association tests that integrate both common and rare genetic variation from putative regulatory elements, including promoters and enhancers for each gene, along with the knockoff-enhanced tests.

License GPL-3

Depends R(>= 3.5.0)

Imports SKAT,
Matrix,
MASS,
WGScan,
SPAtest,
CompQuadForm,
KnockoffScreen

NeedsCompilation no

Repository CRAN

Encoding UTF-8

RoxygenNote 7.1.1

R topics documented:

GeneScan.prelim *The preliminary data management for GeneScan*

Description

This function does the preliminary data management and fit the model under null hypothesis. The output will be used in the other GeneScan functions.

Usage

```
GeneScan.prelim(Y, X = NULL, id = NULL, out_type = "C", B = 1000)
```

Arguments

Y	The outcome variable, an n*1 matrix where n is the total number of observations.
X	An n*d covariates matrix where d is the total number of covariates.
id	The subject id. This is used to match phenotype with genotype. The default is NULL, where the matched phenotype and genotype matrices are assumed.
out_type	Type of outcome variable. Can be either "C" for continuous or "D" for dichotomous. The default is "C".
B	Number of resampling replicates. The default is 1000. A larger value leads to more accurate and stable p-value calculation, but requires more computing time.

Value

It returns a list used for function GeneScan1D() and GeneScan3D().

Examples

```
library(GeneScan3DKnock)

# Load data example
# Y: outcomes, n by 1 matrix where n is the total number of observations
# X: covariates, n by d matrix

data("GeneScan3D.example")
Y=GeneScan3D.example$Y; X=GeneScan3D.example$X;

# Preliminary data management
result.prelim=GeneScan.prelim(Y, X, out_type="C", B=1000)
```

GeneScan1D *Conduct gene-based scan test on the gene buffer region.*

Description

This function conduct gene-based scan test on the gene buffer region using 1D windows under different sizes, do not incorporate any regulatory elements.

Usage

```
GeneScan1D(
  G = G_gene_buffer,
  Z = Z_gene_buffer,
  window.size = c(1000, 5000, 10000),
  pos = pos_gene_buffer,
  MAC.threshold = 5,
  MAF.threshold = 1/sqrt(2 * n),
  Gsub.id = NULL,
  impute.method = "fixed",
  result.prelim = result.prelim
)
```

Arguments

`G`	The genotype matrix in the gene buffer region, which is a n*p matrix where n is the number of subjects and p is the number of genetic variants in the gene buffer region.
`Z`	A p*q genonet matrix matrix where p is the number of genetic variables and q is the number of functional scores (weights). The default is NULL, which uses the beta(MAF; 1,25) weight.
`window.size`	The 1-D window sizes in base pairs to scan the gene buffer region. The default is c(1000,5000,10000).
`pos`	The positions of genetic variants in the gene buffer region, an p dimensional vector. Each position corresponds to a column in the genotype matrix.
`MAC.threshold`	Threshold for minor allele count. Variants below MAC.threshold are ultra-rare variants. The default is 5.
`MAF.threshold`	Threshold for minor allele frequency. Variants below MAF.threshold are rare variants. The default is 1/sqrt(2*n).
`Gsub.id`	The subject id corresponding to the genotype matrix, an n dimensional vector. The default is NULL, where the matched phenotype and genotype matrices are assumed.
`impute.method`	Imputation method when there is missing genotype. Can be "random", "fixed" or "bestguess".
`result.prelim`	The output of function "GeneScan.prelim()".

Value

`GeneScan1D.Cauchy.pvalue`
Cauchy combined p-values under all, common and rare variants for GeneScan1D analysis.

`M` Number of 1D scanning windows.

Examples

```
library(GeneScan3DKnock)

# Load data example
# Y: outcomes, n by 1 matrix where n is the total number of observations
# X: covariates, n by d matrix
# G_gene_buffer: genotype matrix of gene buffer region, n by p matrix
# pos_gene_buffer: positions of genetic variants, p dimentional vector
# Z_gene_buffer: functional annotation matrix, p by q matrix

data(”GeneScan3D.example”)
Y=GeneScan3D.example$Y; X=GeneScan3D.example$X;
G_gene_buffer=GeneScan3D.example$G_gene_buffer;
Z_gene_buffer=GeneScan3D.example$Z_gene_buffer;
pos_gene_buffer=GeneScan3D.example$pos_gene_buffer;
n=length(Y)

# Preliminary data management
result.prelim=GeneScan.prelim(Y, X, out_type=”C”, B=1000)

# Scan the gene buffer region using 1kb, 5kb and 10kb 1-D windows
result.GeneScan1D=GeneScan1D(G=G_gene_buffer,Z=Z_gene_buffer,window.size=c(1000,5000,10000),
pos=pos_gene_buffer,MAC.threshold=5,MAF.threshold=1/sqrt(2*n),
Gsub.id=NULL, impute.method=fixed,result.prelim=result.prelim)
```

`GeneScan3D` *Conduct gene-based scan test on the gene buffer region, adding one promoter and R enhancers.*

Description

This function conduct gene-based scan test on the gene buffer region, incorporating the regulatory elements, i.e., one promoter and R enhancers.

Usage

```
GeneScan3D(
  G = G_gene_buffer,
  Z = Z_gene_buffer,
  G.promoter = G_promoter,
  Z.promoter = Z_promoter,
  G.EnhancerAll = cbind(G_Enhancer1, G_Enhancer2),
  Z.EnhancerAll = rbind(Z_Enhancer1, Z_Enhancer2),
  R = 2,
  p_Enhancer = c(dim(G_Enhancer1)[2], dim(G_Enhancer2)[2]),
  window.size = c(1000, 5000, 10000),
  pos = pos_gene_buffer,
  pos_promoter = pos_promoter,
  MAC.threshold = 5,
  MAF.threshold = 1/sqrt(2 * n),
  Gsub.id = NULL,
  impute.method = "fixed",
  result.prelim = result.prelim
)
```

Arguments

`G` The genotype matrix in the gene buffer region, which is a n*p matrix where n is the number of subjects and p is the number of genetic variants in the gene buffer region.

`Z` A p*q genonet matrix matrix where p is the number of genetic variables and q is the number of functional scores (weights). The default is NULL, which uses the beta(MAF; 1,25) weight.

`G.promoter` The genotype matrix for promoter region.

`Z.promoter` The genonet matrix for promoter region.

`G.EnhancerAll` The genotype matrix for R enhancers, combined together by columns.

`Z.EnhancerAll` The genonet matrix for R enhancers, combined together by rows.

`R` Number of enhancers.

`p_Enhancer` Number of variants in R enhancers, which is a 1*R vector.

`window.size` The 1-D window sizes in base pairs to scan the gene buffer region. The default is c(1000,5000,10000).

`pos` The positions of genetic variants in the gene buffer region, an p dimensional vector. Each position corresponds to a column in the genotype matrix G.

`pos_promoter` The positions of genetic variants in the promoter region. Each position corresponds to a column in the genotype matrix G.promoter.

`MAC.threshold` Threshold for minor allele count. Variants below MAC.threshold are ultra-rare variants. The default is 5.

`MAF.threshold` Threshold for minor allele frequency. Variants below MAF.threshold are rare variants. The default is 1/sqrt(2*n).

`Gsub.id` The subject id corresponding to the genotype matrix, an n dimensional vector. The default is NULL, where the matched phenotype and genotype matrices are assumed.

`impute.method` Imputation method when there is missing genotype. Can be "random", "fixed" or "bestguess".

`result.prelim` The output of function "GeneScan.prelim()".

Value

`GeneScan3D.Cauchy.pvalue`
Cauchy combined p-values under all, common and rare variants for GeneScan3D analysis.

`M` Number of 1D scanning windows.

`minp` Minimum p-values under all, common and rare variants for 3D windows.

RE_minp The regulartory elements in the 3D windows corresponding to the minimum p-values, under all, common and rare variants. 0 represents promoter and a number from 1 to R represents promoter plus r-th enhancer.

Examples

```
library(GeneScan3DKnock)

# Load data example

data("GeneScan3D.example")
Y=GeneScan3D.example$Y; X=GeneScan3D.example$X;
G_gene_buffer=GeneScan3D.example$G_gene_buffer; G_promoter=GeneScan3D.example$G_promoter;
G_Enhancer1=GeneScan3D.example$G_Enhancer1; G_Enhancer2=GeneScan3D.example$G_Enhancer2;
Z_gene_buffer=GeneScan3D.example$Z_gene_buffer; Z_promoter=GeneScan3D.example$Z_promoter;
Z_Enhancer1=GeneScan3D.example$Z_Enhancer1; Z_Enhancer2=GeneScan3D.example$Z_Enhancer2;
pos_gene_buffer=GeneScan3D.example$pos_gene_buffer;
pos_promoter=GeneScan3D.example$pos_promoter;
n=length(Y)

# Preliminary data management
result.prelim=GeneScan.prelim(Y, X, out_type="C", B=1000)

# Conduct 3D gene-based scan test on the gene buffer region, adding one promoter and R enhancers
result.GeneScan3D=GeneScan3D(G=G_gene_buffer,Z=Z_gene_buffer,
G.promoter=G_promoter, Z.promoter=Z_promoter,
G.EnhancerAll=cbind(G_Enhancer1,G_Enhancer2),Z.EnhancerAll=rbind(Z_Enhancer1,Z_Enhancer2),
R=2,p_Enhancer=c(dim(G_Enhancer1)[2],dim(G_Enhancer2)[2]),window.size=c(1000,5000,10000),
pos=pos_gene_buffer,pos_promoter=pos_promoter,MAC.threshold=5,MAF.threshold=1/sqrt(2*n),
Gsub.id=NULL,impute.method=fixed,result.prelim=result.prelim)
```

GeneScan3D.example *Data example for GeneScan3D (A unified framework for gene-based testing with joint analysis of coding and regulatory variation)*

Description

The dataset contains outcome variable Y, covariate X, genotype data for gene buffer region, promoter and two enhancers, weight matrices for functional annotations and positions of genetic variants in gene buffer region as well as promoter.

Usage

```
data("GeneScan3D.example")
```

Format

An object of class `list` of length 12.

Examples

```
data("GeneScan3D.example")

Y=GeneScan3D.example$Y; X=GeneScan3D.example$X

G_gene_buffer=GeneScan3D.example$G_gene_buffer
G_promoter=GeneScan3D.example$G_promoter
G_Enhancer1=GeneScan3D.example$G_Enhancer1
G_Enhancer2=GeneScan3D.example$G_Enhancer2

Z_gene_buffer=GeneScan3D.example$Z_gene_buffer
Z_promoter=GeneScan3D.example$Z_promoter
Z_Enhancer1=GeneScan3D.example$Z_Enhancer1
Z_Enhancer2=GeneScan3D.example$Z_Enhancer2

pos_gene_buffer=GeneScan3D.example$pos_gene_buffer
pos_promoter=GeneScan3D.example$pos_promoter
n=length(Y)
```

GeneScan3DKnock *Integration of knockoff statistics for causal gene identification*

Description

This function calculates the knockoff statistics and q-values after proving the original and knockoff p-values for each gene (or window).

Usage

```
GeneScan3DKnock(
  M = 5,
  p0 = GeneScan3DKnock.example$Cauchy3D.all.original,
  p_ko = cbind(GeneScan3DKnock.example$Cauchy3D.all.ko1,
    GeneScan3DKnock.example$Cauchy3D.all.ko2, GeneScan3DKnock.example$Cauchy3D.all.ko3,
    GeneScan3DKnock.example$Cauchy3D.all.ko4, GeneScan3DKnock.example$Cauchy3D.all.ko5),
  fdr = 0.1,
  gene_id = GeneScan3DKnock.example$gene.id
)
```

Arguments

`M` Number of multiple knockoffs. We use M=5 in our analysis.

`p0` A N-dimensional vector of the original p-values for N genes considered in the analysis. The p-values can be obtained in GeneScan3D() function or other analysis.

`p_ko` A N*M matrix of M knockoff p-values for N genes considered in the analysis. The knockoff p-values can be obtained in R package 'KnockoffScreen'.

`fdr` The false discovery rate (FDR) threshold. The default is 0.1.

`gene_id` The genes id for N genes considered in the analysis, which can also be the windows id or an indicator vector from 1 to N.

Value

W	The knockoff statistics for N genes.
Qvalue	The Q-values for N genes.
gene_sign	Significant genes obtained in the knockoff test with Q-values less then the fdr threshold.

Examples

```
library(GeneScan3DKnock)

# Load data example
data("GeneScan3DKnock.example")

result.GeneScan3DKnock=GeneScan3DKnock(M=5,p0=GeneScan3DKnock.example$Cauchy3D.all.original,
p_ko=cbind(GeneScan3DKnock.example$Cauchy3D.all.ko1,GeneScan3DKnock.example$Cauchy3D.all.ko2,
GeneScan3DKnock.example$Cauchy3D.all.ko3,GeneScan3DKnock.example$Cauchy3D.all.ko4,
GeneScan3DKnock.example$Cauchy3D.all.ko5),fdr = 0.1,gene_id=GeneScan3DKnock.example$gene.id)

#Obtain knockoff statistics, q-values and the significant genes.
W=result.GeneScan3DKnock$W
Qvalue=result.GeneScan3DKnock$Qvalue
gene_sign=result.GeneScan3DKnock$gene_sign
```

GeneScan3DKnock.example

Data example for GeneScan3DKnock (Integration of knockoff statistics for causal gene identification)

Description

This example dataset contains the original and five knockoff p-values for N=100 genes. For each gene, there are gene id, original Cauchy3D p-value and five knockoff Cauchy3D p-values. The data can be used in GeneScan3DKnock() function to calculate the knockoff statistics and q-values for each gene. To generate knockoffs and obtain the knockoff p-values, please find the functions in R Package 'KnockoffScreen'.

Usage

```
data("GeneScan3DKnock.example")
```

Format

An object of class `data.frame` with 100 rows and 7 columns.

Index