Package 'BIGKnock'

February 20, 2022

Type Package

Title BIobank-scale Gene-based association test via Knockoffs

Version 0.1

Author Shiyang Ma, Zihuai He, Iuliana Ionita-Laza

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

Description Functions for knockoff generation of gene and enhancers under biobank-scale data and conduct gene-based association test for related samples under fitted null GLMM.

License GPL-3

Depends R(>= 3.6.0)

Imports SKAT,
Matrix,
MASS,
SPAtest,
CompQuadForm,
irlba

NeedsCompilation no

Repository CRAN

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

R topics documented:

Example.GeneScan3D.UKB.GLMM

Data example for GeneScan3D UKBB gene-based testing based on GLMM.

Description

This example dataset contains a fitted null GLMM results, variance ratio and sparse Sigma matrix, which are obtained by SAIGE/SAIGE-Gene package. Besides, it also contains the original and knockoff genotype for gene and buffer region and enhancer, as well as the positions of gene buffer region.

Usage

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

Format

An object of class `list` of length 8.

Examples

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

result.null.model.GLMM=GeneScan3D.UKB.GLMM.example$result.null.model.GLMM
ratio=GeneScan3D.UKB.GLMM.example$ratio
sparseSigma=GeneScan3D.UKB.GLMM.example$sparseSigma
G_gene_buffer=GeneScan3D.UKB.GLMM.example$G_gene_buffer
pos_gene_buffer=GeneScan3D.UKB.GLMM.example$pos_gene_buffer
G_gene_buffer_knockoff1=GeneScan3D.UKB.GLMM.example$G_gene_buffer_knockoff1
G_enhancer=GeneScan3D.UKB.GLMM.example$G_enhancer
G_enhancer_knockoff1=GeneScan3D.UKB.GLMM.example$G_enhancer_knockoff1
Gsub.id=result.null.model.GLMM$sampleID
```

Example.Knock.generation.enhancer

Data example for knockoff generation of enhancer

Description

This example dataset contains a genotype matrix of enhancer surrounding region and the positions for enhancer. The example dataset is extracted from the 'plinkforGRM_1000samples_10kMarkers' plink data in the extdata folder of SAIGE package, which contains 1000 samples.

Usage

```
data("Knock.generation.enhancer.example")
```

Format

An object of class `list` of length 2.

Examples

```
data(Knock.generation.enhancer.example)
G_enhancer_surround=Knock.generation.enhancer.example$G_enhancer_surround
pos_enhancer=Knock.generation.enhancer.example$pos_enhancer
```

`Example.Knock.generation.gene.buffer` *Data example for knockoff generation of gene buffer region*

Description

This example dataset contains a genotype matrix of gene buffer surrounding region and the positions for gene buffer region. The example dataset is extracted from the 'plinkforGRM_1000samples_10kMarkers' plink data in the extdata folder of SAIGE package, which contains 1000 samples.

Usage

```
data("Knock.generation.gene.buffer.example")
```

Format

An object of class `list` of length 2.

Examples

```
data(Knock.generation.gene.buffer.example)
G_gene_buffer_surround=Knock.generation.gene.buffer.example$G_gene_buffer_surround
pos_gene_buffer=Knock.generation.gene.buffer.example$pos_gene_buffer
```

`GeneScan3D.UKB.GLMM` *Conduct GeneScan3D analysis on UKBB data using fitted null GLMM results.*

Description

This function perform the gene-based test for each gene using the fitted null GLMM and estimated variance ratio obtained from SAIGE/SAIGE-Gene package. For binary traits, we conduct SPA gene-based tests to deal with imbalance case-control issues.

Usage

```
GeneScan3D.UKB.GLMM(
  G = G_gene_buffer,
  G.EnhancerAll = G_EnhancerAll,
  R = length(p_EnhancerAll),
  p_Enhancer = p_EnhancerAll,
  window.size = c(1000, 5000, 10000),
  pos = pos_gene_buffer,
  MAC.threshold = 10,
  MAF.threshold = 0.01,
```

```
  Gsub.id = Gsub.id,
  result.null.model.GLMM = result.null.model.GLMM,
  outcome = "C",
  sparseSigma = sparseSigma,
  ratio = ratio
)
```

Arguments

G	The genotype matrix in the gene buffer region, which is a n*p matrix where n is the number of individuals and p is the number of genetic variants in the gene buffer region.
G.EnhancerAll	The genotype matrix for R enhancers, by combining the genotype matrix of each enhancer by columns.
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 recommended window sizes are 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 and a row in the functional annotation matrix Z.
MAC.threshold	Threshold for minor allele count. Variants below MAC.threshold are ultra-rare variants. The recommended level is 10.
MAF.threshold	Threshold for minor allele frequency. Variants below MAF.threshold are rare variants. The recommended level is 0.01.
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.
result.null.model.GLMM	The fitted null GLMM results obtained from SAIGE/SAIGE-Gene package.
outcome	'C' for quantitative trait, 'D' for binary trait.
sparseSigma	n by n sparse Sigma matrix obtained from SAIGE/SAIGE-Gene package.
ratio	Variance ratio to calibrate test statistics,obtained from SAIGE/SAIGE-Gene package.

Value

GeneScan3D.Cauchy.pvalue	Cauchy combination p-values of all, common and rare variants for GeneScan3D analysis.
M	Number of 1D scanning windows.
minp	Minimum p-values of all, common and rare variants for 3D windows.
RE_minp	The regulartory elements in the 3D windows corresponding to the minimum p-values, for all, common and rare variants. 0 represents promoter and a number from 1 to R represents promoter and r-th enhancer.

Examples

```
data("GeneScan3D.UKB.GLMM.example")
result.null.model.GLMM=GeneScan3D.UKB.GLMM.example$result.null.model.GLMM
ratio=GeneScan3D.UKB.GLMM.example$ratio
sparseSigma=GeneScan3D.UKB.GLMM.example$sparseSigma
G_gene_buffer=GeneScan3D.UKB.GLMM.example$G_gene_buffer
pos_gene_buffer=GeneScan3D.UKB.GLMM.example$pos_gene_buffer
G_gene_buffer_knockoff1=GeneScan3D.UKB.GLMM.example$G_gene_buffer_knockoff1
G_enhancer=GeneScan3D.UKB.GLMM.example$G_enhancer
G_enhancer_knockoff1=GeneScan3D.UKB.GLMM.example$G_enhancer_knockoff1
Gsub.id=result.null.model.GLMM$sampleID

G_EnhancerAll=G_enhancer
p_EnhancerAll=dim(G_enhancer)[2]
G_EnhancerAll_knockoff1=G_enhancer_knockoff1
p_EnhancerAll_knockoff1=dim(G_enhancer_knockoff1)[2]
R=1

result.GeneScan3D_orginal=GeneScan3D.UKB.GLMM(G=G_gene_buffer,
                                              G.EnhancerAll=G_EnhancerAll,
                                              R=R,
                                              p_Enhancer=p_EnhancerAll,
                                              window.size=c(1000,5000,10000),
                                              pos=pos_gene_buffer,
                                              MAC.threshold=10,
                                              MAF.threshold=0.01,
                                              Gsub.id=Gsub.id,
                                              result.null.model.GLMM,
                                              outcome=C,
                                             sparseSigma=sparseSigma,
                                             ratio=ratio)
result.GeneScan3D_orginal$GeneScan3D.Cauchy.pvalue[1]
```

`Knockoffgeneration.enhancer`

Knockoff Generation of enhancer

Description

This function generates multiple knockoff genotypes for enhancer. The knockoff generations are optimized using shrinkage leveraging algorithm.

Usage

```
Knockoffgeneration.enhancer(
  G_enhancer_surround = G_enhancer_surround,
  enhancer_start = enhancer_start,
  enhancer_end = enhancer_start,
  M = 5,
  surround.region = 50000,
  LD.filter = 0.75
)
```

Arguments

G_enhancer_surround	The genotype matrix of the surrounding region for enhancer.
enhancer_start	The start position of enhancer.
enhancer_end	The end position of enhancer.
M	Numer of multiple knockoffs.
surround.region	Surrounding region for enhancer, default is +-50kb.
LD.filter	The correlation threshold for hierarchical clustering. Default LD filter at correlation 0.75

Value

A list file contains original genotype and M knockoff genotypes for enhancer.

Examples

```
library(Matrix)
data(Knock.generation.enhancer.example)
G_enhancer_surround=Matrix(Knock.generation.enhancer.example$G_enhancer_surround)
pos_enhancer=Knock.generation.enhancer.example$pos_enhancer
enhancer_start=min(pos_enhancer)
enhancer_end=max(pos_enhancer)

G_enhancer_knockoff=Knockoffgeneration.enhancer(G_enhancer_surround=G_enhancer_surround,
                                                enhancer_start=enhancer_start,
                                                enhancer_end=enhancer_end,
                                                M=5,surround.region=50000,LD.filter=0.75)
#M=5 knockoffs
G_enhancer_knockoff1=G_enhancer_knockoff[1,,]
G_enhancer_knockoff2=G_enhancer_knockoff[2,,]
G_enhancer_knockoff3=G_enhancer_knockoff[3,,]
G_enhancer_knockoff4=G_enhancer_knockoff[4,,]
G_enhancer_knockoff5=G_enhancer_knockoff[5,,]
```

Knockoffgeneration.gene.buffer

Knockoff Generation of Gene buffer region

Description

This function generates multiple knockoff genotypes for gene buffer region. The knockoff generations are optimized using shrinkage leveraging algorithm.

Usage

```
Knockoffgeneration.gene.buffer(
  G_gene_buffer_surround = G_gene_buffer_surround,
  gene_buffer_start = gene_buffer_start,
  gene_buffer_end = gene_buffer_end,
```

```
  M = 5,
  surround.region = 1e+05,
  LD.filter = 0.75
)
```

Arguments

G_gene_buffer_surround	The genotype matrix of the surrounding region for gene buffer region.
gene_buffer_start	The start position of gene buffer region.
gene_buffer_end	The end position of gene buffer region.
M	Numer of multiple knockoffs.
surround.region	Surrounding region for gene buffer region, default is +-100kb.
LD.filter	The correlation threshold for hierarchical clustering. Default LD filter at correlation 0.75

Value

G_gene_buffer_knockoff
: A list file contains M knockoff genotypes for gene buffer region.

Examples

```
library(Matrix)
data(Knock.generation.gene.buffer.example)
G_gene_buffer_surround=Matrix(Knock.generation.gene.buffer.example$G_gene_buffer_surround)
pos_gene_buffer=Knock.generation.gene.buffer.example$pos_gene_buffer
gene_buffer_start=min(pos_gene_buffer)
gene_buffer_end=max(pos_gene_buffer)

G_gene_buffer_knockoff=Knockoffgeneration.gene.buffer(G_gene_buffer_surround=G_gene_buffer_surround,
                                                      gene_buffer_start=gene_buffer_start,
                                                      gene_buffer_end=gene_buffer_end,
                                                      M=5,surround.region=100000,LD.filter=0.75)
#M=5 knockoffs
G_gene_buffer_knockoff1=G_gene_buffer_knockoff[1,,]
G_gene_buffer_knockoff2=G_gene_buffer_knockoff[2,,]
G_gene_buffer_knockoff3=G_gene_buffer_knockoff[3,,]
G_gene_buffer_knockoff4=G_gene_buffer_knockoff[4,,]
G_gene_buffer_knockoff5=G_gene_buffer_knockoff[5,,]
```

Index