Package: KnockoffScreen (via r-universe)

September 17, 2024

Type Package

Title Whole-Genome Sequencing Data Analysis via Knockoff Statistics

Version 0.3.0

Date 2021-11-19

Author Zihuai He <zihuai@stanford.edu>

Maintainer Zihuai He <zihuai@stanford.edu>

Description Functions for identification of putative causal loci in whole-genome sequencing data. The functions allow genome-wide association scan. It also includes an efficient knockoff generator for genetic data.

License GPL-3

Depends R(>= 3.5.0), Matrix, seqminer, bigmemory

Imports CompQuadForm, data.table, SPAtest, irlba

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-11-19 22:50:02 UTC

Contents

create.MK *Sequential knockoff generator for genetic data*

Description

Generate single/multiple knockoffs for genetic variants for customized analysis.

Usage

```
create.MK(X,pos,M=5,corr_max=0.75,maxN.neighbor=Inf,maxBP.neighbor=100000,
n.AL=floor(10*nrow(X)^(1/3)*log(nrow(X))),thres.ultrarare=25,
R2.thres=1,method='shrinkage',bigmemory=T)
```

Arguments

X	A n*p genotype matrix, where n is the sample size and p is the number of genetic variants.
pos	A vector of length p. Location of the p genetic variants.
M	Number of knockoffs per variant. The default is 5.
corr_max	The correlation threshold for hierarchical clustering, such that variants from two different clusters do not have a correlation greater than corr_max. The hierarchical clustering step is a practical strategy to improve the power for tightly linked variants. The default is 0.75.
maxN.neighbor	The maximum number of neighoring variables used to generate knockoffs. The default is Inf. Smaller number will inprove the computational efficiency, but the knockoffs will be less accurate.
maxBP.neighbor	The size of neighboring region (in base pairs) used to generate knockoffs. The default is 100000.
n.AL	The sample size for the algorithmic leveraging. The default is 10*n^(1/3)*log(n)).
thres.ultrarare	The minor allele count threshold that defines ultrarare variants. The knockoff generation for variants with minor allele counts below the threshold will be based on permutaton. The default is 25.
R2.thres	The maximum R2 allowed in the auto-regressive model. More liberal values (<1) lead to higher power for tightly linked variants, but the knockoffs will be less accurate. The default is 1.
method	The method for subsampling. The default is "shrinkage", corresponding to "shrinkage algorithmic leveraging".
bigmemory	Whether "bigmemory" operation is applied. Default is TRUE.

Value

X_k	An M dimentions list, where each dimention is an n*p matrix as a knockoff copy of original data.

Examples

```
library(KnockoffScreen)

# load example vcf file from package "seqminer"
vcf.filename = system.file("vcf/1000g.phase1.20110521.CFH.var.anno.vcf.gz", package = "seqminer")

## this is how the actual genotype matrix from package "seqminer" looks like
example.G <- t(readVCFToMatrixByRange(vcf.filename, "1:196621007-196716634",annoType='')[[1]])

# filter out constant variants
s<-apply(example.G,2,sd)
example.G<-example.G[,s!=0]
pos<-as.numeric(gsub("^.*:","",colnames(example.G)))

# generate multiple knockoffs
example.G_k<-create.MK(example.G,pos,M=5,corr_max=0.75)
```

KS.chr *Scan a .vcf/.bgen file to identify the putative causal loci in whole-genome sequencing data*

Description

Once the preliminary work is done by "KS.prelim()", this function scan a chromosome given a set of pre-defined windows. It also evalautes invidual variants within those windows.

Usage

```
KS.chr(result.prelim,seq.filename,window.bed,region.pos=NULL,
tested.pos=NULL,excluded.pos=NULL,M=5,thres.single=0.01,
thres.ultrarare=25,thres.missing=0.10,midout.dir=NULL,temp.dir=NULL,
jobtitle=NULL,Gsub.id=NULL,impute.method='fixed',
bigmemory=T,leveraging=T,LD.filter=NULL)
```

Arguments

result.prelim	The output of function "KS.prelim()"
seq.filename	A character specifying the directory (including the file name) of the vcf/bgen file.The algorithm will recognize the file extension and analyze the file accordingly.
window.bed	A matrix specifying the windows being tested. Each row presents a window (chr, start, end), similar to a .bed file. We recommand to define window.bed with window sizes 1000,5000,10000 (base pairs) for sample size ~5000. For studies with smaller sample size, we recommand to increase the window size for stable inference of ultra rare variants.

`region.pos` A vector specifying the start and end of each region being scanned.This provides better control for memory use. Usually we define region.pos such that number of variants in each region is bounded by 5000. For example: region.pos=c(1,100,300) corresponds to two regions (1,100) and (100,300). Default is defined by window.bed.

`tested.pos` A vector specifying position being tested. The default is to test all variants.

`excluded.pos` A vector specifying position being excluded (due to QC or other filters). The default is to not exclude any variant.

`M` Number of knockoffs per variant. The default is 5.

`thres.single` The minor allele frequency threshold to define single genetic variants being tested. Variants with minor allele frequencies above the threshold will be tested individually. The default is 0.01 (for sample size ~5000). Smaller threshold requires a larger sample size.

`thres.ultrarare` The minor allele count threshold to filter out ultrarare variants. Variants with minor allele counts below the threshold will be excluded. The default is 25.

`thres.missing` The missing rate threshold to filter out variants with a low genotyping rate. Variants with missing rate above the threshold will be excluded. The default is 0.1.

`midout.dir` A character specifying the directory to save intermediate results. It is recommended when large scale data is being analyzed.

`temp.dir` A character specifying the directory to save temporary data. It is required when CADD or GenoNet scores are used.

`jobtitle` A character specifying the job title.

`Gsub.id` The subject id corresponding to the genotype matrix, an n dimensional vector. This is used to match phenotype with genotype. The default is NULL, where the subject id in the vcf file is used.

`impute.method` Choose the imputation method when there is missing genotype. Can be "random", "fixed" or "bestguess". Given the estimated allele frequency, "random" simulates the genotype from binomial distribution; "fixed" uses the genotype expectation; "bestguess" uses the genotype with highest probability.

`bigmemory` Whether "bigmemory" operation is applied. Default is TRUE.

`leveraging` Whether "algorithmic leveraging" is applied. Default is TRUE.

`LD.filter` Choose the LD filter for tightly linked variants. Default is 0.75. This applied a hierarchical clustering of genetic variants prior to the analysis, where variants in the same cluster have a pair-wise correlation >=0.75. Then the analysis is restricted to one randomly selected representative variant in each cluster.

Value

`result.window` Results for all windows. Each row presents a window.

`result.single` Results for all individual variants with minor allele frequency above the specified threshold. Each row presents a variant.

Examples

```
library(KnockoffScreen)

# load example vcf file from package "seqminer"
vcf.filename = system.file("vcf/1000g.phase1.20110521.CFH.var.anno.vcf.gz", package = "seqminer")

## this is how the actual genotype matrix from package "seqminer" looks like
example.G <- t(readVCFToMatrixByRange(vcf.filename, "1:196621007-196716634",annoType='')[[1]])

# simulated outcomes, covariates and inidividual id.
Y<-as.matrix(rnorm(nrow(example.G),0,1))
X<-as.matrix(rnorm(nrow(example.G),0,1))
id<-rownames(example.G)

# fit null model
result.prelim<-KS.prelim(Y,X=X,id=id,out_type="C")

# Define the window.bed file
chr<-1
pos.min<-196621007;pos.max<-196716634
window.size=c(2000)

window.bed<-c();
for(size in window.size){
  pos.tag<-seq(pos.min,pos.max,by=size*1/2)
  window.bed<-rbind(window.bed,cbind(chr,pos.tag,pos.tag+size))
}
window.bed<-window.bed[order(as.numeric(window.bed[,2])),]

# scan the vcf file
midout.dir<-NULL # or '/YourProjectDir/MidResults/'
temp.dir<-NULL # or '/YourProjectDir/Temp_out/' #this is a folder to save temporary results
jobtitle<-'YourProjectTitle'

# we set thres.single=0.1,thres.ultrarare=0 for a proof of concept.
# note that the default for real data analysis is thres.single=0.01, thres.ultrarare=25
fit <- KS.chr(result.prelim,vcf.filename,window.bed,M=5,thres.single=0.1,thres.ultrarare=0,
midout.dir=midout.dir,temp.dir=temp.dir,jobtitle=jobtitle)

# summarize the results
result.summary<-KS.summary(fit$result.window,fit$result.single,M=5)
```

KS.prelim *Preliminary data management for KnockoffScreen*

Description

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

Usage

```
KS.prelim(Y, X=NULL, id=NULL, out_type="C")
```

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".

Value

It returns a list that will be passed to function KS.chr().

Examples

```
library(KnockoffScreen)

# load example vcf file from package "seqminer"
vcf.filename = system.file("vcf/1000g.phase1.20110521.CFH.var.anno.vcf.gz", package = "seqminer")

## this is how the actual genotype matrix from package "seqminer" looks like
example.G <- t(readVCFToMatrixByRange(vcf.filename, "1:196621007-196716634",annoType='')[[1]])

# simulated outcomes, covariates and inidividual id.
Y<-as.matrix(rnorm(nrow(example.G),0,1))
X<-as.matrix(rnorm(nrow(example.G),0,1))
id<-rownames(example.G)

# fit null model
result.prelim<-KS.prelim(Y,X=X,id=id,out_type="C")
```

KS.summary *Summarize KnockoffScreen results*

Description

Summarize results generated by function KS.VCF.chr(). Calculate q-values for each window/variant.

Usage

```
KS.summary(result.window,result.single,M)
```

Arguments

result.window	A result matrix generated by KS.VCF.chr() for all windows. Each row present a tested window. If the genome is partitioned into smaller regions and parellel computing is applied (e.g. each chromosome can be partitioned into 50 contiguous regions), the result matrices should be combined and then processed by KS.summary() jointly for genome-wide FDR control.
result.single	A result matrix generated by KS.VCF.chr() for individual variants with minor allele frequency above the specified threshold. Each row present a tested variant. If the genome is partitioned into smaller segments and parellel computing is applied (e.g. each chromosome can be partitioned into 50 contiguous segments), the result matrices should be combined and then processed by KS.summary() jointly for genome-wide FDR control.
M	Number of knockoffs per variant. Should be same as M used in KS.VCF.chr().

Value

result.summary	A matrix summarizing the KnockoffScreen results.

Examples

```
library(KnockoffScreen)

# load example vcf file from package "seqminer"
vcf.filename = system.file("vcf/1000g.phase1.20110521.CFH.var.anno.vcf.gz", package = "seqminer")

## this is how the actual genotype matrix from package "seqminer" looks like
example.G <- t(readVCFToMatrixByRange(vcf.filename, "1:196621007-196716634",annoType='')[[1]])

# simulated outcomes, covariates and inidividual id.
Y<-as.matrix(rnorm(nrow(example.G),0,1))
X<-as.matrix(rnorm(nrow(example.G),0,1))
id<-rownames(example.G)

# fit null model
result.prelim<-KS.prelim(Y,X=X,id=id,out_type="C")

# Define the window.bed file
chr<-1
pos.min<-196621007;pos.max<-196716634
window.size=c(2000)

window.bed<-c();
for(size in window.size){
  pos.tag<-seq(pos.min,pos.max,by=size*1/2)
  window.bed<-rbind(window.bed,cbind(chr,pos.tag,pos.tag+size))
}
window.bed<-window.bed[order(as.numeric(window.bed[,2])),]

# scan the vcf file
midout.dir<-NULL # or '/YourProjectDir/MidResults/'
```

```
temp.dir<-NULL # or '/YourProjectDir/Temp_out/' #this is a folder to save temporary results
jobtitle<-'YourProjectTitle'

# we set thres.single=0.1,thres.ultrarare=0 for a proof of concept.
# note that the default for real data analysis is thres.single=0.01, thres.ultrarare=25
fit <- KS.chr(result.prelim,vcf.filename,window.bed,M=5,thres.single=0.1,thres.ultrarare=0,
midout.dir=midout.dir,temp.dir=temp.dir,jobtitle=jobtitle)

# summarize the results
result.summary<-KS.summary(fit$result.window,fit$result.single,M=5)
```

Index