

# Package ‘KnockoffScreen’

August 11, 2020

**Type** Package

**Title** Whole-Genome Sequencing Data Analysis via Knockoff Statistics

**Version** 0.1.0

**Date** 2020-07-29

**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

**Imports** CompQuadForm, data.table, SPAtest

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-08-11 12:02:07 UTC

## R topics documented:

create.MK . . . . .	2
KS.prelim . . . . .	3
KS.summary . . . . .	4
KS.VCF.chr . . . . .	5
<b>Index</b>	<b>8</b>

---

 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)
```

### Arguments

X	A $n \times p$ 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 <code>corr_max</code> . The hierarchical clustering step is a practical strategy to improve the power for tightly linked variants. The default is 0.75.

### Value

`X_k` An  $M \times n \times p$  array for the  $M$  knockoffs.

### 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)
```

**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", B=1000)
```

**Arguments**

Y	The outcome variable, an $n \times 1$ matrix where $n$ is the total number of observations
X	An $n \times d$ 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 that will be passed to function `KS.VCF.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 individual 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",B=1000)
```

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 parallel 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 parallel 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 individual 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",B=1000)
```

```

# Define the window.bed file
chr<-1
pos.min<-196621007;pos.max<-196716634
window.size=c(1000,5000,10000)

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'

fit <- KS.VCF.chr(result.prelim,vcf.filename>window.bed,M=5,thres.single=0.01,
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.VCF.chr	<i>Scan a VCF file to identify the putative causal loci in whole-genome sequencing data</i>
------------	---

---

## 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.VCF.chr(result.prelim,vcf.filename>window.bed,M=5,thres.single=0.01,
CADD.filename=NULL,GenoNet.filename=NULL,midout.dir=NULL,temp.dir=NULL,
jobtitle=NULL,Gsub.id=NULL,impute.method='fixed')

```

## Arguments

result.prelim	The output of function "KS.prelim()"
vcf.filename	A character specifying the directory (including the file name) of the vcf file.
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.

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.
CADD.filename	A character specifying the directory (including the file name) of CADD score, which can be downloaded at: <a href="https://cadd.gs.washington.edu/download">https://cadd.gs.washington.edu/download</a> , all possible SNVs whole_genome_SNVs.tsv.gz. KnockoffScreen will use the score for improved power. The default is NULL.
GenoNet.filename	A character specifying the directory (including the file name) of tissue/cell type specific GenoNet score, which can be downloaded at: <a href="http://www.funlda.com/genonet/download">http://www.funlda.com/genonet/download</a> . KnockoffScreen will use the score for improved power. The default is NULL.
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.

### 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 individual 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",B=1000)

# Define the window.bed file
chr<-1
pos.min<-196621007;pos.max<-196716634
window.size=c(1000,5000,10000)

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'

fit <- KS.VCF.chr(result.prelim,vcf.filename>window.bed,M=5,thres.single=0.01,
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

- \* **KnockoffGenerator**

- create.MK, [2](#)

- \* **VCF**

- KS.summary, [4](#)

- KS.VCF.chr, [5](#)

- \* **analysis**

- KS.VCF.chr, [5](#)

- \* **preliminary work**

- KS.prelim, [3](#)

- \* **summary**

- KS.summary, [4](#)

create.MK, [2](#)

KS.prelim, [3](#)

KS.summary, [4](#)

KS.VCF.chr, [5](#)