## Package ‘skatMeta’

February 20, 2015

**Type** Package  
**Title** Efficient meta analysis for the SKAT test  
**Version** 1.4.3  
**Date** 2012-11-21  
**Author** Arend Voorman, Jennifer Brody, Thomas Lumley  
**Maintainer** Arend Voorman <voorma@uw.edu>  
**Depends** R (>= 2.10), Matrix, coxme, survival, CompQuadForm  
**Suggests** SKAT, kinship2  
**Description** Computes necessary information to meta analyze SKAT statistics in each individual cohort, and then performs the meta analysis.  
**License** GPL (>= 2)  
**NeedsCompilation** yes  
**Repository** CRAN  
**Date/Publication** 2013-06-04 08:07:42

### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>burdenMeta</td>
<td>2</td>
</tr>
<tr>
<td>singlesnpMeta</td>
<td>4</td>
</tr>
<tr>
<td>skatCohort</td>
<td>6</td>
</tr>
<tr>
<td>skatCohortAdj</td>
<td>9</td>
</tr>
<tr>
<td>skatExample</td>
<td>11</td>
</tr>
<tr>
<td>skatMeta</td>
<td>12</td>
</tr>
<tr>
<td>skatOMeta</td>
<td>15</td>
</tr>
<tr>
<td>SNPInfo</td>
<td>18</td>
</tr>
</tbody>
</table>

**Index** 20
burdenMeta  Combine burden tests from multiple cohorts

Description

Takes as input 'skatCohort' objects (from the skatCohort function), and meta-analyzes the corresponding burden test.

Usage

burdenMeta(..., SNPInfo=NULL, wts = 1, snpNames = "Name", aggregateBy = "gene", mafRange = c(0,0.5), verbose=FALSE)

Arguments

...  skatCohort objects
SNPInfo  the SNP Info file. This should contain 'Name' and 'gene' fields, which match the 'Name' and 'gene' fields of the SNP Info file used in each cohort. Only SNPs and genes in this table will be meta analyzed, so this may be used to restrict the analysis.
wts  weights for the burden test, as a function of maf, or a character string specifying weights in the SNP Info file.
snpNames  The field of SNPInfo where the SNP identifiers are found. Default is 'Name'
aggregateBy  The field of SNPInfo on which the skat results were aggregated. Default is 'gene'. For single snps which are intended only for single variant analyses, it is recomended that they have a unique identifier in this field.
mafRange  Range of MAF’s to include in the analysis (endpoints included). Default is all SNPs (0 <= MAF <= 0.5).
verbose  logical. whether progress bars should be printed.

Details

This function uses the scores and their variances available in a skatCohort to perform burden tests. Though coefficients are reported, the tests are formally score tests, and the coefficients can be thought of as one-step approximations to those reported in a Wald test.

Value

a data frame with the following columns:
gene  the name of the gene or unit of aggregation being meta analyzed
p  the p-value from the burden tests
beta  approximate coefficient for the effect of genotype
se  approximate standard error for the effect of genotype
cmaftotal  the cumulative minor allele frequency of the gene
The cumulative minor allele frequency of SNPs used in the analysis

The number of SNPs in the gene

The number of SNPs used in the analysis

The number of 'missing' SNPs. For a gene with a single SNP this is the number of individuals which do not contribute to the analysis, due to cohorts that did not report results for that SNP. For a gene with multiple SNPs, is totalled over the gene.

Author(s)
Arie Voorman, Jennifer Brody

See Also
skatMeta skatMeta skatCohort

Examples

```r
## Load example data for two cohorts
data(skatExample)

## Run on each cohort:
coh1 <- skatCohort(Z = Z1, y = 1, SNPInfo = SNPInfo, data = pheno1)
coh2 <- skatCohort(Z = Z2, y = 1, SNPInfo = SNPInfo, data = pheno2)

## Combine results:
out <- burdenMeta(coh1, coh2, SNPInfo = SNPInfo, mafRange = c(0.01))
head(out)

## Not run:
## Compare with analysis on full data set:
bigZ <- matrix(NA, 2*n, nrow(SNPInfo))
colnames(bigZ) <- SNPInfo$Name
for (gene in unique(SNPInfo$gene)) {
  snpNnames <- SNPInfo$Name[SNPInfo$gene == gene]
  bigZ[,n] <- SNPInfo$gene == gene[, snp.names %in% colnames(Zn)] <- Zn[, na.omit(match(snp.names, colnames(Zn)))]
  bigZ[2:n,] <- SNPInfo$gene == gene[, snp.names %in% colnames(Zn)] <- Zn[, na.omit(match(snp.names, colnames(Zn)))]
}
pheno <- rbind(pheno1[, c("y", "sex", "bmi")], pheno2[, c("y", "sex", "bmi")])
burden.p <- c(by(SNPInfo$Name, SNPInfo$gene, function(snp.names) {
  inds <- match(snp.names, colnames(bigZ))
  burden <- rowSums(bigZ[, na.omit(inds)], na.rm = TRUE)
  mod <- lm(y ~ burden + gl(2, nrow(pheno1)), data = pheno)
  summary(mod)$coef[2, 4]
})),
head(cbind(out$p, burden.p))
```
singlesnpMeta

Meta analyze single snp effects from multiple cohorts

Description

Takes as input `skatCohort` objects (from the `skatCohort` function), and meta analyzes them.

Usage

`singlesnpMeta(..., SNPInfo=NULL, snpNames = "Name", aggregateBy = "gene", cohortBetats = TRUE, verbose = FALSE)`

Arguments

- ... skatCohort objects
- SNPInfo The SNP Info file. This should contain the fields listed in snpNames and aggregateBy. Only SNPs in this table will be meta analyzed, so this may be used to restrict the analysis.
- snpNames The field of SNPInfo where the SNP identifiers are found. Default is 'Name'
- aggregateBy The field of SNPInfo on which the skat results were aggregated. Default is 'gene'. Though gene groupings are not explicitly required for single snp analysis, it is required to find where single snp information is stored in the skatCohort objects.
- cohortBetats Whether or not to include cohort-level effects in the output.
- verbose logical. Whether progress bars should be printed.

Details

This function meta analyzes score tests for single variant effects. Though the test is formally a score test, coefficients and standard errors are also returned, which can be interpreted as a 'one-step' approximation to the maximum likelihood estimates.

Value

a data frame with the gene, snp name, meta analysis.

Author(s)

Arie Voorman, Jennifer Brody
References


See Also

skatCohort skatMeta skatOMeta

Examples

data(skatExample)

### run on each cohort:
cohort1 <- skatCohort(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo, data = pheno1)
cohort2 <- skatCohort(Z=Z2, y~sex+bmi, SNPInfo = SNPInfo, data = pheno2)

### combine results:
out <- singlesnpMeta(cohort1, cohort2, SNPInfo = SNPInfo)
head(out)

# Not run:
#compare
bigZ <- matrix(NA,2*n,nrow(SNPInfo))
colnames(bigZ) <- SNPInfo$Name
for(gene in unique(SNPInfo$gene)){
  snpZnames <- SNPInfo$Name[SNPInfo$gene == gene][,
  snp.names %in% colnames(Z1)] <- Z1[, na.omit(match(snp.names,colnames(Z1)))]
  bigZ[(i+1):1:(2*n),SNPInfo$gene == gene][,
  snp.names %in% colnames(Z2)] <- Z2[, na.omit(match(snp.names,colnames(Z2)))]
}
pheno <- rbind(pheno1[,c("y","sex","bmi")],pheno2[,c("y","sex","bmi")])
out3 <- apply(bigZ,2,function(z){
  if(any(!is.na(z))){
    z[is.na(z)] <- mean(z,na.rm=TRUE)
    mod <- lm(y ~ sex+bmi+c(rep(1,n),rep(0,n))+z,data=pheno)
    beta <- sqrt(vcov(mod)["z","z")]
    se <- sqrt(vcov(mod)["z","z")]
    p <- pchisq((beta/se)^2,df=1,lower=F)
    return(c(beta,se,p))
  } else {
    return(c(0,Inf,1))
  }
})
out3 <- t(out3[,match(out$Name,colnames(out3))])

#plot
par(mfrow=c(2,2))
plot(x=out3[,1],y=out$beta, xlab = "complete data (Wald)",
  ylab = "meta-analysis (Score)", main = "coefficients");abline(0,1)
Run SKAT on data from a single cohort.

Description
This function computes and organizes the necessary output to efficiently meta-analyze a linear model SKAT and other tests. Note that the SKAT test is not computed by these functions. The output must be passed to one of skatMeta, burdenMeta, or singlesnpMeta.

Unlike the SKAT package which operates on one gene at a time, these functions are intended to operate on many genes, e.g. a whole exome, to facilitate meta analysis of whole genomes or exomes.

Usage
skatCohort(Z, formula, family = gaussian(), SNPInfo=NULL, snpNames = "Name", aggregateBy = "gene", data=parent.frame(), verbose = FALSE)

skatFamCohort(Z, formula, SNPInfo=NULL, snpNames = "Name", aggregateBy = "gene", data=parent.frame(), fullkins, sparse = TRUE, verbose = FALSE)

skatCoxCohort(Z, formula, SNPInfo=NULL, snpNames = "Name", aggregateBy = "gene", data=parent.frame(), verbose = FALSE)

Arguments

Z A genotype matrix (dosage matrix) - rows correspond to individuals and columns correspond to SNPs. Use 'NA' for missing values. The column names of this matrix should correspond to SNP names in the SNP information file.

formula Base formula, of the kind used in glm() - typically of the form y~covariate1 + covariate2. For Cox models, the formula follows that of the coxph() function.

family for skatCohort: either gaussian(), for continuous data, or binomial() for 0/1 outcomes. Binary outcomes are not currently supported for family data.

SNPInfo SNP Info file - must contain fields given in 'snpName' and 'aggregateBy'.

.snpNames The field of SNPInfo where the SNP identifiers are found. Default is 'Name'

.aggregateBy The field of SNPInfo on which the skat results were aggregated. Default is 'gene'. For single snps which are intended only for single variant analyses, it is recommended that they have a unique identifier in this field.

data data frame in which to find variables in the formula

verbose logical. whether or not to print the progress bar.
fullkinds

for skatFamCohort: the kinship matrix. See lmkem and the kinship2 package for more details
sparse

for skatFamCohort: whether or not to use a sparse Matrix approximation for dense kinship matrices (defaults to TRUE)

Details

This function computes the necessary information to meta analyze SKAT analyses: the individual SNP scores, their MAF, and a covariance matrix for each unit of aggregation. Note that the SKAT test is *not* calculated by this function. The output must be passed to one of skatMeta, burdenMeta, or singlesnpMeta.

A crucial component of SKAT and other region-based tests is a common unit of aggregation across cohorts. This is given in the SNP information file (argument SNPInfo), which pairs SNPs to a unit of aggregation (typically a gene). The additional arguments Name and aggregateBy specify the columns of the SNP information file which contain these pairings. Note that the column names of the genotype matrix Z must match the names given in the Name field.

Using skatCohort, users are strongly recommended to use all SNPs, even if they are monomorphic in your study. This is for two reasons; firstly, monomorphic SNPs provide information about MAF across all studies; without providing the information we are unable to tell if a missing SNP data was monomorphic in a cohort, or simply failed to genotype adequately in that cohort. Second, even if some SNPs will be filtered out of a particular meta-analysis (e.g., because they are intronic or common) constructing skatCohort objects describing all SNPs will reduce the workload for subsequent follow-up analyses.

Note: to view results for a single cohort, one can pass a single skatCohort object to a function for meta-analysis.

Value

an object of class 'skatCohort'. This is a list, not meant for human consumption, but to be fed to skatMeta() or another function. The names of the list correspond to gene names. Each element in the list contains

scores The scores (y-yhat)^t g
cov The variance of the scores. When no covariates are used, this is the LD matrix.
n The number of subjects
maf The minor allele frequency
sey The residual standard error.

Note

For skatCoxCohort, the signed likelihood ratio statistic is used instead of the score, as the score test is anti-conservative for proportional hazards regression. The code for this routine is based on the coxph.fit function from the survival package.

Please see the package vignette for more details.

Author(s)

Arie Voorman, Jennifer Brody
References


Chen H, Meigs JB, Dupuis J. Sequence Kernel Association Test for Quantitative Traits in Family Samples. Genetic Epidemiology. (To appear)


See Also

skatMeta burdenMeta singlesnpMeta skatOMeta coxph

Examples

```r
###load example data for two cohorts:
### see ?skatExample
data(skatExample)

###run on each cohort:
cohort1 <- skatCohort(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo, data = phenol)
cohort2 <- skatFamCohort(Z=Z2, y~sex+bmi, SNPInfo = SNPInfo, 
 fullkinds=kins, data=pheno2)

### combine results:
#skat
out <- skatMeta(cohort1, cohort2, SNPInfo = SNPInfo)
head(out)

###T1 test
out.t1 <- burdenMeta(cohort1, cohort2, SNPInfo = SNPInfo, mafRange = c(0,0.01))
head(out.t1)

###single snp tests:
out.ss <- singlesnpMeta(cohort1, cohort2, SNPInfo = SNPInfo)
head(out.ss)

## Not run:
################################
###binary data

cohort1 <- skatCohort(Z=Z1, ybin~1, family=binomial(), SNPInfo = SNPInfo, data = phenol)
out <- skatMeta(cohort1, SNPInfo = SNPInfo)
head(out)

################################
###survival data

cohort1 <- skatCoxCohort(Z=Z1, Surv(time,status)-strata(sex)+bmi, SNPInfo = SNPInfo, data = phenol)
```
skatCohortAdj

Run SKAT on data from a single cohort, conditional on specified SNP effects

Description

This function works exactly as skatCohort, but with the additional argument ‘adjustments’ specifying genes for which conditional analyses are desired, and which SNPs to condition on.

Usage

skatCohortAdj(Z, formula, family = gaussian(), SNPInfo=NULL, adjustments= NULL, snpNames = "Name", aggregateBy = "gene", data=parent.frame())

skatFamCohortAdj(Z, formula, SNPInfo=NULL, adjustments= NULL, snpNames = "Name", aggregateBy = "gene", fullkins, sparse = TRUE, data=parent.frame())

Arguments

Z A genotype matrix (dosage matrix) - rows correspond to individuals and columns correspond to SNPs. Use 'NA' for missing values. The column names of this matrix should correspond to SNP names in the SNP information file.

formula Base formula, of the kind used in glm() - typically of the form y~covariate1 + covariate2. For Cox models, the formula follows that of the coxph() function.

family for skatCohort: either gaussian(), for continuous data, or binomial() for 0/1 outcomes. Binary outcomes are not currently supported for family data.

SNPInfo SNP Info file - must contain fields given in 'snpName' and 'aggregateBy'.

adjustments A data frame of the same format at SNPInfo, pairing genes to analyze with snp

snpNames The field of SNPInfo where the SNP identifiers are found. Default is 'Name'

aggregateBy The field of SNPInfo on which the skat results were aggregated. Default is 'gene'. For single snps which are intended only for single variant analyses, it is recommended that they have a unique identifier in this field.

data data frame in which to find variables in the formula

fullkins for skatFamCohort: the kinship matrix. See lmekin and the kinship2 package for more details

sparse for skatFamCohort: whether or not to use a sparse Matrix approximation for dense kinship matrices (defaults to TRUE)
Details

This function has the same syntax as \texttt{skatCohort} and \texttt{skatFamCohort}, but requires an extra argument ‘adjustments’. This is a data frame of the same format as SKAT meta, i.e. with a ‘snpNames’ and ‘aggregateBy’ columns. The function works by looping through the genes in the adjustment file, adding the corresponding SNPs to the null model in SKAT. For instance, if one wants to adjust ‘gene1’ for SNPs a and b (which need not be in gene 1), and ‘gene2’ for SNPs c, the adjustments would be something like

\begin{verbatim}
adjustments <- data.frame(Name = c("a", "b", "c"),
                          gene = c("gene1", "gene1", "gene2")
\end{verbatim}

See the examples for an illustration.

Value

an object of class 'skatCohort'. Note that unlike output from the function \texttt{skatCohort}, the null models in each element of the list may be different. When meta analyzing these, it may be good to subset the SNPInfo file to the genes of interest.

Author(s)

Arie Voorman, Jennifer Brody

See Also

\texttt{skatCohort skatMeta burdenMeta singlesnpMeta coxph}

Examples

```r
### load example data for two cohorts:
### see ?skatExample
data(skatExample)

class(skatExample)

data <- skatExample
data <- data[1:3, 20:100]

# specify adjustment variables
adjustments <- SNPInfo[,1:3]

### run on each cohort:
cohort1.adj <- skatCohortAdj(data, y~sex+bmi, SNPInfo = SNPInfo,
                              adjustments=adjustments)
cohort2.adj <- skatFamCohortAdj(data, y~sex+bmi, SNPInfo = SNPInfo,
                                 adjustments=adjustments)
SNPInfo.sub <- subset(SNPInfo, !is.na(SNPInfo$gene &
                          !is.na(SNPInfo$Name &
                                 !is.na(SNPInfo$adjusted$gene)))

# test
out.skat <- skatMeta(cohort1.adj, cohort2.adj, SNPInfo = SNPInfo.sub)
head(out.skat)

# test
out.tl <- burdenMeta(cohort1.adj, cohort2.adj, SNPInfo = SNPInfo.sub,
                     mafRange = c(0.01))
head(out.tl)
```
## Description
Contains simulated data for two cohorts. See the example for the exact code used to generate the data.

## Usage
```
data(skatExample)
```

## Format
This contains simulated data for two cohorts to illustrate skatMeta package.
- $Z_1, Z_2$: Genotype matrices for cohorts 1 and 2 respectively.
- `pheno1, pheno2`: Phenotype matrices for cohorts 1 and 2 respectively.
- `kins`: The kinship matrix for cohort 2.

## Examples
```
# Data generated by
## Not run:
skExample <- skatExample
set.seed(20)
N <- 600 # Observations per cohort
d <- 2000 # SNPs
k <- 100 # Genes

#### First cohort of unrelated individuals:
Z1 <- replicate(d, rbinom(N, 2, rbeta((N), 3, 200)))

## assign SNP id's to the columns
colnames(Z1) <- sample(d + 50, d) + 1e6

pheno1 <- data.frame("y" = rnorm(N), "sex" = rep(1:2, (N/2)), "bmi" = rnorm(N, 25, 2),
"ybin" = rbinom(N, 1, .5), "time" = rpois(N, 5), "status" = rbinom(N, 1, .9))

genes <- paste0("gene", 1:k)
SNPInfo <- data.frame("Name" = 1:(d + 50) + 1e6, "gene" = sort(sapply(genes, d + 50, replace=T)))

#### Second cohort of family data:
# 150 families of size 4
```
require(kinship)
fullped<-data.frame(famid=rep(1:(n/4),each=4),id=10001:(10000+n),fa=rep(0,n),mo=rep(0,n))
fullped$fal[(1:(n/4))*4-1]<-fullped$fal[(1:(n/4))*4]<-1:(n/4)*4+9997
fullped$mo[(1:(n/4))*4-1]<-fullped$mo[(1:(n/4))*4]<-1:(n/4)*4+9998
kins = makekinship(fullped$famid, fullped$id, fullped$fa, fullped$mo)

## generate a phenotype with 20% 'heritability':
pheno2<-data.frame("id"=10001:(10000+n),"y"=t(rnorm(n))%*% chol(0.2*2*as.matrix(kins) +
0.8*diag(n)),"sex"=rep(1:2,(n/2)),"bmi"=rnorm(n,25,2))
Z2 <- replicate(d, rbinom(n,2,rbeta((n/4),3,200)[fullped$famid]))
colnames(Z2) <- sample(d+50,d) + 1e6

## End(Not run)

---

**skatMeta**

*Combine SKAT analyses from one or more cohorts*

**Description**

Takes as input 'skatCohort' objects (from e.g. skatCohort), and meta analyzes them.

**Usage**

```r
skatMeta(..., SNPInfo=NULL, wts = function(maf){dbeta(maf,1,25)}, method = "saddlepoint",
snpNames = "Name", aggregateBy = "gene", mafRange = c(0,0.5), verbose=FALSE)
```

**Arguments**

- `...` skatCohort objects
- `SNPInfo` the SNP Info file. This should contain 'Name' and 'gene' fields, which match the 'Name' and 'gene' fields of the SNP Info file used in each cohort. Only SNPs and genes in this table will be meta analyzed, so this may be used to restrict the analysis.
- `wts` Either a function to calculate testing weights, or a character specifying a vector of weights in the SNPInfo file. For skatMeta the default are the 'beta' weights.
- `method` p-value calculation method. Default is 'saddlepoint', 'integration' is the Davies method used in the SKAT package. See pchisqsum() for more details.
- `snpNames` The field of SNPInfo where the SNP identifiers are found. Default is 'Name'
- `aggregateBy` The field of SNPInfo on which the skat results were aggregated. Default is 'gene'. For single snps which are intended only for single variant analyses, it is recommanded that they have a unique identifier in this field.
- `mafRange` Range of MAF’s to include in the analysis (endpoints included). Default is all SNPs (0 <= MAF <= 0.5).
- `verbose` logical. Whether or not to print progress bars.
**Details**

skatMeta implements an efficient SKAT meta analysis by meta-analyzing scores statistics and their variances.

Note: all cohorts must use coordinated SNP Info files - that is, the SNP names and gene definitions must be the same.

Please see the package vignette for more details.

**Value**

a data frame with columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gene</td>
<td>Name of the gene.</td>
</tr>
<tr>
<td>p</td>
<td>p-value of the SKAT test.</td>
</tr>
<tr>
<td>Q</td>
<td>The SKAT Q-statistic, defined as ( \sum_j w_j S_j ), where ( S_j ) is the squared score for SNP ( j ), and ( w_j ) is a weight.</td>
</tr>
<tr>
<td>cmaf</td>
<td>The cumulative minor allele frequency.</td>
</tr>
<tr>
<td>nmiss</td>
<td>The number of ‘missing’ SNPs. For a gene with a single SNP this is the number of individuals which do not contribute to the analysis, due to cohorts that did not report results for that SNP. For a gene with multiple SNPs, is totalled over the gene.</td>
</tr>
<tr>
<td>nsnps</td>
<td>The number of SNPs in the gene.</td>
</tr>
</tbody>
</table>

**Author(s)**

Arie Voorman, Jennifer Brody

**References**


**See Also**

skatCohort burdenMeta singlesnpMeta skatOMeta

**Examples**

```r
### load example data for 2 cohorts
data(skatExample)

### run on each cohort:
cohort1 <- skatCohort(Z=Z1, y=sex+bmi, SNPInfo = SNPInfo, data=pheno1)
cohort2 <- skatFamCohort(Z=Z2, y=sex+bmi, SNPInfo = SNPInfo, fullkins=kinds, data=pheno2)

### combine results:
#skat
out <- skatMeta(cohort1, cohort2, SNPInfo = SNPInfo)
```
head(out)

# T1 test
out.t1 <- burdenMeta(cohort1, cohort2, SNPInfo = SNPInfo, mafRange = c(0, 0.01))
head(out.t1)

## single SNP tests:
out.ss <- singleSnpMeta(cohort1, cohort2, SNPInfo = SNPInfo)
head(out.ss)

### binary data
cohort1 <- skatCohort(Z = Z1, ybin = 1, family = binomial(), SNPInfo = SNPInfo, data = pheno1)
out.bin <- skatMeta(cohort1, SNPInfo = SNPInfo)
head(out.bin)

### survival data
cohort1 <- skatCoxCohort(Z = Z1, Surv(time, status) ~ strata(sex) + bmi, SNPInfo = SNPInfo, data = pheno1)
out.surv <- skatMeta(cohort1, SNPInfo = SNPInfo)
head(out.surv)

## Not run:
### Compare with SKAT on full data set
require(SKAT)
n <- nrow(pheno1)
bigZ <- matrix(NA, 2 * n, nrow(SNPInfo))
colnames(bigZ) <- SNPInfo$name
for (gene in unique(SNPInfo$gene)){
snp.names <- SNPInfo$name[SNPInfo$gene == gene]
bigZ[1:n, SNPInfo$gene == gene][, snp.names %in% colnames(Z1)] <- Z1[, na.omit(match(snp.names, colnames(Z1)))]
bigZ[(n+1):2*n, SNPInfo$gene == gene][, snp.names %in% colnames(Z2)] <- Z2[, na.omit(match(snp.names, colnames(Z2)))]}
pheno <- rbind(pheno1[, c("y", "bmi", "sex")], pheno2[, c("y", "bmi", "sex")])
obj <- SKAT_Null_Model(y = sex + bmi + gl(2, nrow(pheno1)), data = pheno)
skat.pkg.p <- c(by(SNPInfo$Name, SNPInfo$gene, function(snp.names){
inds <- match(snp.names, colnames(bigZ))
if(sum(!is.na(inds)) == 0) return(1)
SKAT(bigZ[, na.omit(inds)], obj, is_check = TRUE, missing = 1) + p.value})
head(cbind(out$p, skat.pkg.p))

# Note: SKAT ignores family structure, resulting in p-values that are systematically too small:
plot(y = out$p, x = skat.pkg.p, ylab = "SKAT meta p-values", xlab = "SKAT p-values")
**skatOMeta**

Combine SKAT-O analyses from one or more cohorts.

**Description**

Takes as input `skatCohort` objects (from e.g. `skatCohort`), and meta analyzes them, using SKAT-O. See the package vignette for more extensive documentation.

**Usage**

```
skatOMeta(..., SNPInfo=NULL, skat.wts = function(maf)(dbeta(maf,1,25)), burden.wts = function(maf)(as.numeric(maf <= 0.01) ), rho=c(0,1), method = c("integration", "saddlepoint", "liu"), snpNames = "Name", aggregateBy = "gene", mafRange = c(0,0.5), verbose=FALSE)
```

**Arguments**

- `...` skatCohort objects
- `SNPInfo` the SNP Info file. This should contain 'Name' and 'gene' fields, which match the 'Name' and 'gene' fields of the SNP Info file used in each cohort. Only SNPs and genes in this table will be meta analyzed, so this may be used to restrict the analysis.
- `skat.wts` Either a function to calculate testing weights for SKAT, or a character specifying a vector of weights in the SNPInfo file. For skatOMeta the default are the 'beta' weights.
- `burden.wts` Either a function to calculate weights for the burden test, or a character specifying a vector of weights in the SNPInfo file. For skatOMeta the default are the T1 weights.
- `rho` A sequence of values that specify combinations of SKAT and a burden test to be considered. Default is c(0,1), which considers SKAT and a burden test.
- `method` p-value calculation method. Should be one of 'saddlepoint', 'integration', or 'liu'.
- `snpNames` The field of SNPInfo where the SNP identifiers are found. Default is 'Name'
aggregateBy  The field of SNPInfo on which the skat results were aggregated. Default is 'gene'. For single snps which are intended only for single variant analyses, it is recomended that they have a unique identifier in this field.

mafRange  Range of MAF's to include in the analysis (endpoints included). Default is all SNPs (0 <= MAF <= 0.5).

verbose  logical. Whether or not to print progress bars.

Details

skatOMeta() implements the SKAT-Optimal test, which picks the ‘best’ combination of SKAT and a burden test, and then corrects for the flexibility afforded by this choice. Specifically, if the SKAT statistic is Q1, and the squared score for a burden test is Q2, SKAT-O considers tests of the form (1-rho)*Q1 + rho*Q2, where rho between 0 and 1. The values of rho are specified by the user using the argument rho. In the simplest form, which is the default, SKAT-O computes a SKAT test and a T1 test, and reports the minimum p-value, corrected for multiple testing. See the vignette or the accompanying references for more details.

If there is a single variant in the gene, or the burden test is undefined (e.g. there are no rare alleles for the T1 test), SKAT is reported (i.e. rho=0).

Note 1: the SKAT package uses the same weights for both SKAT and the burden test, which this function does not.

Note 2: all cohorts must use coordinated SNP Info files - that is, the SNP names and gene definitions must be the same.

Note 3: The method of p-value calculation is much more important here than in SKAT. The ‘integration’ method is fast and typically accurate for p-values larger than 1e-9. The saddlepoint method is slower, but has higher relative accuracy.

Note 4: Since p-value calculation can be slow for SKAT-O, and less accurate for small p-values, a reasonable alternative would be to first calculate SKAT and a burden test, and record the minimum p-value, which is a lower bound for the SKAT-O p-value. This can be done quickly and accurately. Then, one would only need to perform SKAT-O on the small subset of genes that are potentially interesting.

Please see the package vignette for more details.

Value

a data frame with columns:

gene  Name of the gene.
p  p-value of the SKAT-O test.
pmin  The minimum of the p-values considered by SKAT-O (not corrected for multiple testing!).
rho  The value of rho which gave the smallest p-value.
cmaf  The cumulative minor allele frequency.
nmiss  The number of ‘missing’ SNPs. For a gene with a single SNP this is the number of individuals which do not contribute to the analysis, due to cohorts that did not report results for that SNP. For a gene with multiple SNPs, is totalled over the gene.
nsnps  The number of SNPs in the gene.
errflag  An indicator of possible error: 0 suggests no error, > 0 indicates probable loss of accuracy.

Author(s)
Arie Voorman, Jennifer Brody

References

See Also
   skatMeta skatCohort burdenMeta singlesnpMeta

Examples

## Not run:
### load example data for 2 cohorts
data(skatExample)

### run on each cohort:
cohort1 <- skatCohort(Z=Z1, y=sex+bmi, SNPInfo = SNPInfo, data =pheno1)
cohort2 <- skatCohort(Z=Z2, y=sex+bmi, SNPInfo = SNPInfo,
fullkins=kins, data =pheno2)

### combine results:
### skat-0 with default settings:
out1 <- skatOMeta(cohort1, cohort2, SNPInfo = SNPInfo, method = "int")
head(out1)

### skat-0, using a large number of combinations between SKAT and T1 tests:
out2 <- skatOMeta(cohort1, cohort2, rho = seq(0,1,length=11),
SNPInfo = SNPInfo, method = "int")
head(out2)

# rho = 0 indicates SKAT gave the smaller p-value (or the T1 is undefined)
# rho=1 indicates the burden test was chosen
# 0 < rho < 1 indicates some other value was chosen
#notice that most of the time either the SKAT or T1 is chosen
table(out2$rho)

### skat-0 with beta-weights used in the burden test:
out3 <- skatOMeta(cohort1,cohort2, burden.wts = function(maf){dbeta(maf,1,25) },
rho=seq(0,1,length=11),SNPInfo = SNPInfo, method="int")
head(out3)
table(out3$rho)

############################################
## binary data

cohort1 <- skatCohort(Z=Z1, ybin~1, family=binomial(), SNPInfo = SNPInfo, data = pheno1)
out.bin <- skatOMeta(cohort1, SNPInfo = SNPInfo, method="int")
head(out.bin)

############################################
## survival data

cohort1 <- skatCoxCohort(Z=Z1, Surv(time,status)-strata(sex)+bmi, SNPInfo = SNPInfo, data = pheno1)
out.surv <- skatOMeta(cohort1, SNPInfo = SNPInfo, method="int")
head(out.surv)

## Compare with SKAT and T1 tests on their own:
cohort1 <- skatCohort(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo, data = pheno1)
cohort2 <- skatFamCohort(Z=Z2, y~sex+bmi, SNPInfo = SNPInfo, fullkinds=kinds, id=pheno2$id, data=pheno2)

out.skat <- skatMeta(cohort1, cohort2, SNPInfo=SNPInfo)
out.t1 <- burdenMeta(cohort1, cohort2, wts= function(maf)(as.numeric(maf <= 0.01)), SNPInfo=SNPInfo)

# plot results
# We compare the minimum p-value of SKAT and T1, adjusting for multiple tests
# using the Sidak correction, to that of SKAT-O.
#
par(mfrow=c(1,3))
pseq <- seq(0,1,length=100)
plot(y=out.skat$p, x=out1$p,xlab="SKAT-O p-value", ylab="SKAT p-value", main ="SKAT-O vs SKAT")
lines(y=pseq,x=1-(1-pseq)^2,col=2,lty=2, lwd=2)
abline(0,1)
plot(y=out.t1$p, x=out1$p,xlab="SKAT-O p-value", ylab="T1 p-value", main ="SKAT-O vs T1")
lines(y=pseq,x=1-(1-pseq)^2,col=2,lty=2, lwd=2)
abline(0,1)
plot(y=min(out.t1$p, out.skat$p,na.rm=T), x=out1$p,xlab="SKAT-O p-value", ylab="min(T1,SKAT) p-value", main ="min(T1,SKAT) vs SKAT-O")
lines(y=pseq,x=1-(1-pseq)^2,col=2,lty=2, lwd=2)
abline(0,1)
legend("bottomright", lwd=2,lty=2,col=2,legend="Bonferroni correction")

## End(Not run)
Description

Contains standard Names and associated genes for the Illumina HumanExome BeadChip

Usage

data(SNPInfo)

Format

A data frame with 247504 observations on the following 2 variables.

<table>
<thead>
<tr>
<th>Chr</th>
<th>Chromosome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>factor: illumina variant names</td>
</tr>
<tr>
<td>SKATgene</td>
<td>factor: gene names</td>
</tr>
</tbody>
</table>

Details

There are several non-exonic SNPs included. For these SNPs the 'gene' name is the same as the illumina variant name.

References


Examples

data(SNPInfo)

## summary of the data set:
summary(as.numeric(table(SNPInfo$SKATgene)))
hist(table(SNPInfo$SKATgene), nclass = 300, xlim = c(0, 50),
main = "SNPs per gene", xlab = "#SNPs", ylab = "#Genes")
Index

*Topic \textasciitilde kwd1
  skatCohortAdj, 9
*Topic \textasciitilde kwd2
  skatCohortAdj, 9
*Topic datasets
  skatExample, 11
  SNPInfo, 19

burdenMeta, 2, 6–8, 10, 13, 17
coxph, 8, 10
kins (skatExample), 11
pheno1 (skatExample), 11
pheno2 (skatExample), 11

singlesnpMeta, 4, 6–8, 10, 13, 17
skatCohort, 2–5, 6, 9, 10, 12, 13, 15, 17
skatCohortAdj, 9
skatCoxCohort (skatCohort), 6
skatExample, 11
skatFamCohort, 10
skatFamCohort (skatCohort), 6
skatFamCohortAdj (skatCohortAdj), 9
skatMeta, 3, 5–8, 10, 12, 17
skatOMeta, 3, 5, 8, 13, 15
SNPInfo, 18

Z1 (skatExample), 11
Z2 (skatExample), 11