Package 'seqMeta'

February 9, 2017

Type Package

Title Meta-Analysis of Region-Based Tests of Rare DNA Variants

Version 1.6.7

Date 2016-09-24

Author Arie Voorman, Jennifer Brody, Han Chen, Thomas Lumley, Brian Davis

Maintainer Brian Davis <Brian.Davis281@gmail.com>

URL https://github.com/DavisBrian/seqMeta

BugReports https://github.com/DavisBrian/seqMeta/issues

Depends R (>= 3.1.0), methods, survival

Imports Matrix, coxme (>= 2.2-4), CompQuadForm

Suggests SKAT, kinship2, testthat

Description Computes necessary information to meta analyze region-based tests for rare genetic variants (e.g. SKAT, T1) in individual studies, and performs meta analysis.

License GPL (>= 2)

NeedsCompilation yes

RoxygenNote 5.0.1

Repository CRAN

Date/Publication 2017-02-09 15:24:14

R topics documented:

`burdenMeta` *Meta analyze burden tests from multiple studies*

Description

Takes as input 'seqMeta' objects (from the `prepScores` 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

`...`	`seqMeta` 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.
`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'. Though gene groupings are not explicitely required for single snp analysis, it is required to find where single snp information is stored in the seqMeta objects.
`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 seqMeta object 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
cmafUsed	the cumulative minor allele frequency of snps used in the analysis
nsnpsTotal	the number of snps in the gene
nsnpsUsed	the number of snps used in the analysis
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 studies 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 skatOMeta prepScores

Examples

```
###load example data for two studies:
### see ?seqMetaExample
data(seqMetaExample)

####run on each cohort:
cohort1 <- prepScores(Z=Z1, y~1, SNPInfo=SNPInfo, data=pheno1)
cohort2 <- prepScores(Z=Z2, y~1, SNPInfo=SNPInfo, data=pheno2)

#### combine results:
out <- burdenMeta(cohort1, cohort2, 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$gene)) {
   snp.names <- SNPInfo$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","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))

#will be slightly different:
plot(y=out$p,x=burden.p, ylab = "burden meta p-values", xlab = "complete data p-values")

## End(Not run)
```

---

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

Description

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

Usage

```
prepCondScores(Z, formula, family = stats::gaussian(), SNPInfo = NULL,
  adjustments = NULL, snpNames = "Name", aggregateBy = "gene",
  kins = NULL, 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	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'. See Details.
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.
kins	the kinship matrix for related individuals. Only supported for family=gaussian(). See lmekin in the kinship2 package for more details.
sparse	whether or not to use a sparse Matrix approximation for dense kinship matrices (defaults to TRUE).
data	data frame in which to find variables in the formula

Details

This function has the same syntax as prepCondScores, but requires an extra argument 'adjustments'. This is a data frame of the same format as the SNPInfo, 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. For instance, if one wants to adjuste 'gene1' for SNPs a and b (which need not be in gene 1), and 'gene2' for SNPs c, the adjustments would be something like `adjustments = data.frame(Name = c("a","b","c"), gene = c("gene1","gene1","gene2"))`

See the examples for an illustration.

Value

an object of class 'seqMeta'. Note that unlike output from the function prepScores, 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

prepScores skatMeta burdenMeta singlesnpMeta

Examples

```
###load example data for two studies:
### see ?seqMetaExample
data(seqMetaExample)

#specify adjustment variables
adjustments <- SNPInfo[c(1:3, 20,100), ]
adjustments

####run on each study:
cohort1.adj <- prepCondScores(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo,
                adjustments=adjustments, data =pheno1)
cohort2.adj <- prepCondScores(Z=Z2, y~sex+bmi, SNPInfo = SNPInfo,
                adjustments=adjustments, kins=kins, data=pheno2)

SNPInfo.sub <- subset(SNPInfo, (SNPInfo$gene \%in\% adjustments$gene) &
                                !(SNPInfo$Name \%in\% adjustments$Name) )

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

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

##single snp tests:
```

```
out.ss <- singlesnpMeta(cohort1.adj,cohort2.adj, SNPInfo = SNPInfo.sub)
head(out.ss)
```

---

prepCox *Prepare scores for region based (meta) analysis*

Description

This function computes and organizes the neccesary output to efficiently meta-analyze SKAT and other tests. Note that the tests are *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

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

prepScores(Z, formula, family = stats::gaussian(), SNPInfo = NULL,
  snpNames = "Name", aggregateBy = "gene", kins = NULL, sparse = TRUE,
  data = parent.frame(), verbose = FALSE)

prepScoresX(Z, formula, male, family = stats::gaussian(), SNPInfo = NULL,
  snpNames = "Name", aggregateBy = "gene", kins = NULL, sparse = TRUE,
  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.
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'. See Details.
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.
data	data frame in which to find variables in the formula
verbose	logical. whether or not to print the progress bar.
family	either gaussian(), for continuous data, or binomial() for 0/1 outcomes. Binary outcomes are not currently supported for family data.

kins	the kinship matrix for related individuals. Only supported for family=gaussian(). See lmekin in the kinship2 package for more details.
sparse	whether or not to use a sparse Matrix approximation for dense kinship matrices (defaults to TRUE).
male	For analyzing the X chromosome, with prepScoresX, 'male' is the gender (0/1 or F/T) indicating female/male. See details.

Details

This function computes the neccesary 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 accross studies. This is given in the SNP information file (argument SNPInfo), which pairs SNPs to a unit of aggregation (typically a gene). The additional arguments snpNames 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 snpNames field.

Using prepScores, 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 study, or simply failed to genotype adequately in that study. Second, even if some SNPs will be filtered out of a particular meta-analysis (e.g., because they are intronic or common) constructing seqMeta objects describing all SNPs will reduce the workload for subsequent follow-up analyses.

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

Value

an object of class 'seqMeta'. 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 alternate allele frequency
sey	The residual standard error.

Note

For prepCox, 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

Wu, M.C., Lee, S., Cai, T., Li, Y., Boehnke, M., and Lin, X. (2011) Rare Variant Association Testing for Sequencing Data Using the Sequence Kernel Association Test (SKAT). American Journal of Human Genetics.

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

Lin, DY and Zeng, D. On the relative efficiency of using summary statistics versus individual-level data in meta-analysis. Biometrika. 2010.

See Also

skatMeta burdenMeta singlesnpMeta skatOMeta coxph

Examples

```
###load example data for two studies:
### see ?seqMetaExample
data(seqMetaExample)

####run on each cohort:
cohort1 <- prepScores(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo, data =pheno1)
cohort2 <- prepScores(Z=Z2, y~sex+bmi, SNPInfo = SNPInfo, kins=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 <- prepScores(Z=Z1, ybin~1, family=binomial(), SNPInfo = SNPInfo, data =pheno1)
out <- skatMeta(cohort1, SNPInfo = SNPInfo)
head(out)

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

```
## End(Not run)
```

---

prepScores2 *Prepare scores for region based (meta) analysis*

Description

This function is a replacement for prepScores, prepScoresX and prepCox. It computes and organizes the neccesary output to efficiently meta-analyze SKAT and other tests. Note that the tests are *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

```
prepScores2(Z, formula, family = "gaussian", SNPInfo = NULL,
  snpNames = "Name", aggregateBy = "gene", kins = NULL, sparse = TRUE,
  data = parent.frame(), male = NULL, 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	either 'gaussian', for continuous data, 'binomial' for 0/1 outcomes or 'cox' for survival models. Family data not currently supported for binomial or survival outcomes. Male also not supported for survival outcomes. See Details.
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'. See Details.
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.
kins	the kinship matrix for related individuals. Only supported for family=gaussian(). See lmekin in the kinship2 package for more details.
sparse	whether or not to use a sparse Matrix approximation for dense kinship matrices (defaults to TRUE).
data	data frame in which to find variables in the formula
male	For analyzing the X chromosome, with prepScoresX, 'male' is the gender (0/1 or F/T) indicating female/male. See details.
verbose	logical. whether or not to print the progress bar.

Details

This function is a drop in replacement for prepScores, prepScoresX, and prepCox. If family is 'cox' then the call is equivalent to prepCox and an error will occur if either male or kins is provided. When family is 'gaussian' or 'binomial' and male is not provided then the call is equivalent to prepScores. Whereas if male is provided then the call is equivalent to prepScoresX.

This function computes the neccesary 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 accross studies. This is given in the SNP information file (argument `SNPInfo`), which pairs SNPs to a unit of aggregation (typically a gene). The additional arguments `snpNames` 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 `snpNames` field.

Using `prepScores2`, 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 study, or simply failed to genotype adequately in that study. Second, even if some SNPs will be filtered out of a particular meta-analysis (e.g., because they are intronic or common) constructing seqMeta objects describing all SNPs will reduce the workload for subsequent follow-up analyses.

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

Value

an object of class 'seqMeta'. 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 alternate allele frequency
`sey`	The residual standard error.

Note

For survival models, 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)

Brian Davis, Arie Voorman, Jennifer Brody

References

Wu, M.C., Lee, S., Cai, T., Li, Y., Boehnke, M., and Lin, X. (2011) Rare Variant Association Testing for Sequencing Data Using the Sequence Kernel Association Test (SKAT). American Journal of Human Genetics.

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

Lin, DY and Zeng, D. On the relative efficiency of using summary statistics versus individual-level data in meta-analysis. Biometrika. 2010.

See Also

prepScores prepScoresX prepCox skatMeta burdenMeta singlesnpMeta skatOMeta

Examples

```
###load example data for two studies:
### see ?seqMetaExample
data(seqMetaExample)

####run on each cohort:
cohort1 <- prepScores2(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo, data = pheno1)
cohort2 <- prepScores2(Z=Z2, y~sex+bmi, SNPInfo = SNPInfo, kins = 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 <- prepScores2(Z=Z1, formula = ybin~1, family = "binomial",
                       SNPInfo = SNPInfo, data = pheno1)
out <- skatMeta(cohort1, SNPInfo = SNPInfo)
head(out)

####################
####survival data
cohort1 <- prepScores2(Z=Z1, formula = Surv(time,status)~strata(sex)+bmi,
                       family = "cox", SNPInfo = SNPInfo, data = pheno1)
out <- skatMeta(cohort1, SNPInfo = SNPInfo)
head(out)

## End(Not run)
```

seqMeta *seqMeta: Meta-Analysis of Region-Based Tests of Rare DNA Variants*

Description

Computes necessary information to meta analyze region-based tests for rare genetic variants (e.g. SKAT, T1) in individual studies, and performs meta analysis.

Details

To learn more about seqMeta, start with the vignettes: `browseVignettes(package = "seqMeta")`

seqMetaExample *Example data for illustrating seqMeta*

Description

Contains simulated data for two cohorts. See the example for the exact code used to generate the data.

Usage

```
data(seqMetaExample)
```

Format

This contains simulated data for two cohorts to illustrate seqMeta package

`Z1`,`Z2` 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:
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"=as.character(1:(d+50) + 1e6),
"gene"=sort(sample(genes,d+50,replace=T)), stringsAsFactors=FALSE)
#SNPInfo <- data.frame("Name" =1:(d+50) + 1e6, "gene" = sort(sample(genes,d+50,replace=T)))

#####Second cohort of family data:
# 150 families of size 4
require(kinship2)
fullped<-data.frame(famid=rep(1:(n/4),each=4),id=10001:(10000+n),fa=rep(0,n),mo=rep(0,n))
fullped$fa[(1:(n/4))*4-1]<-fullped$fa[(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)
```

`singlesnpMeta` *Meta analyze single snp effects from multiple studies*

Description

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

Usage

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

Arguments

`...`	`seqMeta` 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 explicitely required for single snp analysis, it is required to find where single snp information is stored in the seqMeta objects.

studyBetas	Whether or not to include study-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

Lin, DY and Zeng, D. On the relative efficiency of using summary statistics versus individual-level data in meta-analysis. Biometrika. 2010.

See Also

prepScores burdenMeta skatMeta skatOMeta

Examples

```
###load example data for two studies:
### see ?seqMetaExample
data(seqMetaExample)

####run on each study:
cohort1 <- prepScores(Z=Z1, y~sex+bmi, SNPInfo = SNPInfo, data =pheno1)
cohort2 <- prepScores(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)) {
   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","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 <- mod$coef["z"]
           se <- sqrt(vcov(mod)["z", "z"])
           p <- pchisq( (beta/se)^2,df=1,lower=F)
           return(c(beta,se,p))
         } else {
           return(c(0,Inf,1))
         }
 })
 out3 <- t(out3[,match(out$Name,colnames(out3))])

 ##plot
 par(mfrow=c(2,2))
 plot(x=out3[,1],y=out$beta, xlab="complete data (Wald)",
      ylab="meta-analysis (Score)", main="coefficients"); abline(0,1)
 plot(x=out3[,2],y=out$se, xlab="complete data (Wald)",
      ylab="meta-analysis (Score)", main="standard errors"); abline(0,1)
 plot(x=out3[,3],y=out$p, xlab="complete data (Wald)",
      ylab="meta-analysis (Score)", main="p-values"); abline(0,1)

## End(Not run)
```

---

skatMeta *Combine SKAT analyses from one or more studies*

Description

Takes as input 'seqMeta' objects (from the prepScores function), and meta-analyzes them.

Usage

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

Arguments

...	seqMeta 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.
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'. Though gene groupings are not explicitely required for single snp analysis, it is required to find where single snp information is stored in the seqMeta objects.
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

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

Note: all studies 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 the following columns:

gene	the name of the gene or unit of aggregation being meta analyzed
p	p-value of the SKAT test.
Q	The SKAT Q-statistic, defined as sum_j w_jS_j, where S_j is the squared score for SNP j, and w_j is a weight.
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 studies 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.

Author(s)

Arie Voorman, Jennifer Brody

References

Wu, M.C., Lee, S., Cai, T., Li, Y., Boehnke, M., and Lin, X. (2011) Rare Variant Association Testing for Sequencing Data Using the Sequence Kernel Association Test (SKAT). American Journal of Human Genetics.

See Also

prepScores burdenMeta singlesnpMeta skatOMeta

Examples

```
###load example data for two studies:
### see ?seqMetaExample
data(seqMetaExample)

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

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

## Not run:
##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 <- prepScores(Z=Z1, ybin~1, family=binomial(), SNPInfo=SNPInfo, data=pheno1)
out.bin <- skatMeta(cohort1, SNPInfo=SNPInfo)
head(out.bin)

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

##### 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","sex","bmi")], pheno2[,c("y","sex","bmi")])
```

```
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 strucutre, 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")
abline(0,1)

ignore family structure:
cohort1 <- prepScores(Z=Z1, y~sex+bmi, SNPInfo=SNPInfo, data=pheno1)
cohort2 <- prepScores(Z=Z2, y~sex+bmi, SNPInfo=SNPInfo, data=pheno2)

out.nofam <- skatMeta(cohort1,cohort2,SNPInfo=SNPInfo)
plot(y=out.nofam$p,x=skat.pkg.p, ylab = "SKAT meta p-values", xlab = "SKAT p-values")
abline(0,1)

## End(Not run)
```

---

skatOMeta *Combine SKAT-O analyses from one or more studies.*

Description

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

Usage

```
skatOMeta(..., SNPInfo = NULL, skat.wts = function(maf) {
  stats::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

...	seqMeta 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.
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'. Though gene groupings are not explicitely required for single snp analysis, it is required to find where single snp information is stored in the seqMeta objects.
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

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 studies 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 the following columns:

gene	Name of the gene or unit of aggregation being meta analyzed

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 studies 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

Wu, M.C., Lee, S., Cai, T., Li, Y., Boehnke, M., and Lin, X. (2011) Rare Variant Association Testing for Sequencing Data Using the Sequence Kernel Association Test (SKAT). American Journal of Human Genetics.

Lee, S. and Wu, M.C. and Lin, X. (2012) Optimal tests for rare variant effects in sequencing association studies. Biostatistics.

See Also

skatOMeta prepScores burdenMeta singlesnpMeta

Examples

```
## Not run:
### load example data for 2 studies
data(seqMetaExample)

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

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

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

####################
####survival data
cohort1 <- prepCox(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 <- prepScores(Z=Z1, y~sex+bmi, SNPInfo=SNPInfo, data=pheno1)
cohort2 <- prepScores(Z=Z2, y~sex+bmi, SNPInfo=SNPInfo, kins=kins, 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=pmin(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)
```

SNPInfo *Illumina HumanExome BeadChip SNP Information file*

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.

`Chr` Chromosome

`Name` factor: illumina variant names

`SKATgene` factor: gene names

Details

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

References

Grove ML, Cochran BJ, Haritunians T, Bis JC, Taylor KD, Hansen M, O'Donnell CJ, Rotter JI, Boerwinkle E, CHARGE Exome Chip Genotyping Committee; Best practices and joint calling of the Illumina HumanExome BeadChip: the CHARGE consortium; (Abstract/Program #1445W). Presented at the 62nd Annual Meeting of The American Society of Human Genetics (ASHG), November 7, 2012, San Francisco, CA.

Grove ML, Yu B, Cochran BJ, Haritunians T, Bis JC, Taylor KD, Hansen M, Borecki I, Cupples LA, Fornage M, Gudnason V, Harris T, Katherisan S, Kraaij R, Launer LJ, Levy D, Liu Y, Mosley T, Peloso GM, Psaty BM, Rich SS, Rivadeneira F, Siscovick DS, Smith AV, Uitterlinden A, van Duijn CM, Wilson JG, O'Donnell CJ, Rotter JI, Boerwinkle E. Best practices and joint calling of the Illumina HumanExome BeadChip: the CHARGE consortium. PLoS One [submitted]

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