Package 'SAIGEgds'

November 12, 2026

Type Package

Title Scalable Implementation of Generalized mixed models using GDS files in Phenome-Wide Association Studies

Version 2.12.0

Date 2026-04-23

Depends R (>= 4.0.0), gdsfmt (>= 1.28.0), SeqArray (>= 1.50.2), Rcpp

LinkingTo Rcpp, RcppArmadillo, RcppParallel (>= 5.0.0)

Imports methods, stats, utils, Matrix, RcppParallel, SKAT, CompQuadForm, survey

Suggests parallel, markdown, rmarkdown, crayon, SNPRelate, RUnit, knitr, ggmanh, BiocGenerics

Description Scalable implementation of generalized mixed models with highly optimized C++ implementation and integration with Genomic Data Structure (GDS) files. It is designed for single variant tests and set-based aggregate tests in large-scale Phenome-wide Association Studies (PheWAS) with millions of variants and samples, controlling for sample structure and case-control imbalance. The implementation is based on the SAIGE R package (v0.45, Zhou et al. 2018 and Zhou et al. 2020), and it is extended to include the state-of-the-art ACAT-O set-based tests. Benchmarks show that SAIGEgds is significantly faster than the SAIGE R package. Optional OpenCL-based GPU acceleration is supported for the GRM cross-product computation in null model fitting and for GRM construction.

License GPL-3

SystemRequirements GNU make

VignetteBuilder knitr

ByteCompile TRUE

URL https://github.com/AbbVie-ComputationalGenomics/SAIGEgds

biocViews Software, Genetics, StatisticalMethod, GenomeWideAssociation

git_url https://git.bioconductor.org/packages/SAIGEgds

git_branch RELEASE_3_23

git_last_commit d1163ac

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Xiuwen Zheng [aut, cre] (ORCID:
<https://orcid.org/0000-0002-1390-0708>),
Wei Zhou [ctb] (the original author of the SAIGE R package),
J. Wade Davis [ctb]

Maintainer Xiuwen Zheng <xiuwen.zheng@abbvie.com>

Contents

SAIGEgds-package 2
glmmHeritability 4
pACAT 5
seqAssocGLMM_ACAT_O 6
seqAssocGLMM_ACAT_V 9
seqAssocGLMM_Burden 11
seqAssocGLMM_SKAT 14
seqAssocGLMM_SPA 16
seqFitLDpruning 19
seqFitNullGLMM_SPA 21
seqFitSparseGRM 25
seqGetGenoByGeno 27
seqSAIGE_LoadPval 28

Index **30**

`SAIGEgds-package` *Scalable Implementation of Generalized mixed models in Phenome-Wide Association Studies using GDS files*

Description

Scalable and accurate implementation of generalized mixed mode with the support of Genomic Data Structure (GDS) files and highly optimized C++ implementation. It is designed for single variant tests in large-scale phenome-wide association studies (PheWAS) with millions of variants and hundreds of thousands of samples, e.g., UK Biobank genotype data, controlling for case-control imbalance and sample structure in single variant association studies.

The implementation of SAIGEgds is based on the original SAIGE R package (v0.29.4.4) [Zhou et al. 2018] https://github.com/weizhouUMICH/SAIGE/releases/tag/v0.29.4.4. All of the calculation with single-precision floating-point numbers in SAIGE are replaced by the double-precision calculation in SAIGEgds. SAIGEgds also implements some of the SPAtest functions in C to speed up the calculation of Saddlepoint Approximation.

Details

Package:	SAIGEgds
Type:	Package
License:	GPL version 3

Author(s)

Xiuwen Zheng <xiuwen.zheng@abbvie.com>, Wei Zhou (the original author of the SAIGE R package, https://github.com/weizhouUMICH/SAIGE)

References

Zheng X, Davis J.Wade. SAIGEgds – an efficient statistical tool for large-scale PheWAS with mixed models. *Bioinformatics* (2020). DOI: 10.1093/bioinformatics/btaa731.

Zhou W, Nielsen JB, Fritsche LG, Dey R, Gabrielsen ME, Wolford BN, LeFaive J, VandeHaar P, Gagliano SA, Gifford A, Bastarache LA, Wei WQ, Denny JC, Lin M, Hveem K, Kang HM, Abecasis GR, Willer CJ, Lee S. Efficiently controlling for case-control imbalance and sample relatedness in large-scale genetic association studies. *Nat Genet* (2018). Sep;50(9):1335-1341.

Zhou W, Zhao Z, Nielsen JB, Fritsche LG, LeFaive J, Taliun SAG, Bi W, Gabrielsen ME, Daly MJ, Neale BM, Hveem K, Abecasis GR, Willer CJ, et al. Scalable generalized linear mixed model for region-based association tests in large biobanks and cohorts. Nat Genet. 2020; 52: 634-9.

Zheng X, Gogarten S, Lawrence M, Stilp A, Conomos M, Weir BS, Laurie C, Levine D. SeqArray – A storage-efficient high-performance data format for WGS variant calls. *Bioinformatics* (2017). DOI: 10.1093/bioinformatics/btx145.

Examples

```
# open the GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, trait.type="binary",
    geno.sparse=FALSE)

# p-value calculation
assoc <- seqAssocGLMM_SPA(gdsfile, glmm, mac=10)

head(assoc)
```

```
# close the GDS file
seqClose(gdsfile)
```

`glmmHeritability` *Heritability estimation*

Description

Get the heritability estimate from the SAIGE model.

Usage

```
glmmHeritability(modobj, adjust=TRUE)
```

Arguments

`modobj`	an R object for SAIGE model parameters
`adjust`	if `TRUE` and binary outcomes, uses adjusted tau estimate for the heritability estimation

Details

In SAIGE, penalized quasi-likelihood (PQL) is used to estimate the variance component parameter tau. It is known to produce biased estimate of the variance component tau using PQL. If `adjust=TRUE` for binary outcomes, tau is adjusted based prevalence and observed tau using the data in Supplementary Table 7 (Zhou et al. 2018) to reduce the bias of PQL estimate of variance component.

Value

Return a liability scale heritability.

Author(s)

Xiuwen Zheng

See Also

`seqFitNullGLMM_SPA`

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
```

```
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, trait.type="binary")

glmmHeritability(glmm)

seqClose(gdsfile)
```

---

pACAT *Cauchy Combination Test*

Description

P-value calculation from Cauchy combination test.

Usage

```
pACAT(p, w=NULL)
pACAT2(p, maf, wbeta=c(1,25))
```

Arguments

p	a numeric vector for p-values
w	weight for each p-value
maf	minor allele frequency for each p-value
wbeta	weights for per-variant effect, using beta distribution `dbeta()` according to variant's MAF

Value

Return a single number for the combined p-value.

References

Liu Y., Cheng S., Li Z., Morrison A.C., Boerwinkle E., Lin X.; ACAT: A Fast and Powerful p Value Combination Method for Rare-Variant Analysis in Sequencing Studies. Am J Hum Genetics 104, 410-421 (2019).

See Also

seqFitNullGLMM_SPA, seqAssocGLMM_SPA

Examples

```
p1 <- 10^-4
p2 <- 10^-5
p3 <- 10^-(3:20)
sapply(p3, function(p) pACAT(c(p1, p2, p)))

pACAT2(c(10^-4, 10^-6), c(0.01, 0.005))
```

`seqAssocGLMM_ACAT_O` *ACAT-O tests*

Description

Set-based ACAT-O tests using mixed models.

Usage

```
seqAssocGLMM_ACAT_O(gdsfile, modobj, units, maxMAF=0.01, wbeta=AggrParamBeta,
    missing=0.05, collapse.mac=10, collapse.method=c("max", "sum"),
    ccimb.adj=TRUE, ER.mac=4.5, dsnode="",
    geno.model=c("additive", "dominant", "recessive"),
    res.savefn="", res.compress="ZIP",
    parallel=FALSE, verbose=TRUE, verbose.maf=FALSE)
seqAssocGLMM_ACAT_O_GT(gt_lst, modobj, maxMAF=0.01, wbeta=AggrParamBeta,
    missing=0.05, collapse.mac=10, collapse.method=c("max", "sum"),
    ccimb.adj=TRUE, ER.mac=4.5,
    geno.model=c("additive", "dominant", "recessive"),
    verbose=TRUE, verbose.maf=FALSE)
```

Arguments

`gdsfile`	a SeqArray GDS filename, or a GDS object
`modobj`	an R object for SAIGE model parameters
`units`	a list of units of selected variants, with S3 class `"SeqUnitListClass"` defined in the SeqArray package
`maxMAF`	a numeric vector for MAF thresholds, e.g., `c(0.01, 0.001)` for rare variants, or `c(0.5, 0.1, 0.01)` for both common and rare variants
`wbeta`	weights for per-variant effect, using beta distribution `dbeta()` according to variant's MAF; a length-two vector, or a matrix with two rows for multiple beta parameters; by default, using beta(1,1) and beta(1,25) both
`missing`	missing threshold for variants (checking <= `missing`), `NaN` for no filter
`collapse.mac`	a threshold of minor allele count for collapsing ultra rare variants used in ACAT-V and SKAT tests if `mac` <= `collapse.mac`, 10 by default
`collapse.method`	`"max"` (by default), `"sum"`

ccimb.adj	whether adjusting for case-control imbalance
ER.mac	the maximum sum of MAC for the variants with efficient resampling p-values
dsnode	"" for automatically searching the GDS nodes "genotype" and "annotation/format/DS", or use a user-defined GDS node in the file
geno.model	a character string for the genetic model: "additive" (default), "dominant" or "recessive"; the dominant model recodes dosages as min(G, 1), and the recessive model recodes as max(G-1, 0)
res.savefn	an RData or GDS file name, "" for no saving
res.compress	the compression method for the output file, it should be one of LZMA, LZMA_RA, ZIP, ZIP_RA and none
parallel	FALSE (serial processing), TRUE (multicore processing), a numeric value for the number of cores, or other value; parallel is passed to the argument cl in seqParallel, see seqParallel for more details
gt_lst	a list of numeric genotype matrices (sample by variant) or Matrix objects, where rows correspond to the samples in the model and columns to variants; each list element is a unit
verbose	if TRUE, show information
verbose.maf	if TRUE, show summary of MAFs in units

Details

seqAssocGLMM_ACAT_O() reads genotype data from a GDS file and supports parallel computation. seqAssocGLMM_ACAT_O_GT() takes a list of in-memory genotype matrices gt_lst directly, which is useful when genotypes are already loaded or constructed outside of the GDS framework. The rows of each matrix must match the samples used in the null model.

seqUnitFilterCond() in the SeqArray package can be used to restrict the variant sets units to a range of MAC and/or MAF. ACAT-O combines the p-values from ACAT-V, burden and SKAT together to calculate the final p-value via Cauchy distribution.

For more details of the ACAT-O method, please refer to the ACAT paper [Liu et al. 2019] (see the reference section).

When geno.model="dominant", the dosages are recoded as min(G, 1) after allele flipping to the minor allele, testing whether having at least one copy of the minor allele is associated with the trait. When geno.model="recessive", dosages are recoded as max(G-1, 0), testing whether having two copies of the minor allele is associated. MAF/MAC filtering and the reported allele frequencies use the original (additive) genotype values regardless of the genetic model.

Value

Return a data.frame with the following components if not saving to a file:

chr	chromosome;
start	the starting position;
end	the ending position;
maxMAF	maximum of MAFs;

numvar	the number of variants in the window; if weight="Cauchy", it is the number of individual p-values in the Cauchy test
macmin	the minimum of MAC in the window;
macmed	the median of MAC in the window;
macmax	the maximum of MAC in the window;
summac	the sum of minor allele counts;
weight	the weighting strategy with the beta distribution parameters
pval	ACAT-O p-value;
n_collapse	the number of ultra rare variants for collapsing burden tests (<= collapse.mac);
p.burden	burden tests;
p.skat	SKAT tests;
p.acatv	ACAT-V tests;
burden.beta	beta coefficient of burden test;
burden.se	SE of burden beta coefficient;

Author(s)

Xiuwen Zheng

References

Liu Y., Chen S., Li Z., Morrison A.C., Boerwinkle E., Lin X. ACAT: A Fast and Powerful p Value Combination Method for Rare-Variant Analysis in Sequencing Studies. Am J Hum Genetics 104, 410-421 (2019).

See Also

seqAssocGLMM_ACAT_V, seqAssocGLMM_Burden, seqAssocGLMM_SKAT, seqUnitFilterCond

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

sp_grm_fn <- system.file("extdata", "grm1k_10k_sp_grm.rds", package="SAIGEgds")
sp_grm <- readRDS(sp_grm_fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, sp_grm,
    trait.type="binary")
```

```
# get a list of variant units for burden tests
units <- seqUnitSlidingWindows(gdsfile, win.size=50, win.shift=250)

a <- seqAssocGLMM_ACAT_O(gdsfile, glmm, units, maxMAF=c(0.1,0.01))
head(a)

# close the GDS file
seqClose(gdsfile)
```

`seqAssocGLMM_ACAT_V` *ACAT-V tests*

Description

Set-based ACAT-V tests using mixed models.

Usage

```
seqAssocGLMM_ACAT_V(gdsfile, modobj, units, maxMAF=0.01, wbeta=AggrParamBeta,
    missing=0.05, ccimb.adj=TRUE, collapse.mac=10, ER.mac=4.5, dsnode="",
    geno.model=c("additive", "dominant", "recessive"),
    res.savefn="", res.compress="ZIP", parallel=FALSE, verbose=TRUE,
    verbose.maf=FALSE)
```

Arguments

`gdsfile`	a SeqArray GDS filename, or a GDS object
`modobj`	an R object for SAIGE model parameters
`units`	a list of units of selected variants, with S3 class `"SeqUnitListClass"` defined in the SeqArray package
`maxMAF`	a numeric vector for MAF thresholds, e.g., `c(0.01, 0.001)` for rare variants, or `c(0.5, 0.1, 0.01)` for both common and rare variants
`wbeta`	weights for per-variant effect, using beta distribution `dbeta()` according to variant's MAF; a length-two vector, or a matrix with two rows for multiple beta parameters; by default, using beta(1,1) and beta(1,25) both
`missing`	missing threshold for variants (checking <= `missing`), `NaN` for no filter
`ccimb.adj`	whether adjusting for case-control imbalance or not
`collapse.mac`	a threshold of minor allele count for collapsing ultra rare variants if `mac` <= `collapse.mac`, 10 by default
`ER.mac`	the maximum sum of MAC for the variants with efficient resampling p-values
`dsnode`	"" for automatically searching the GDS nodes "genotype" and "annotation/format/DS", or use a user-defined GDS node in the file
`geno.model`	a character string for the genetic model: `"additive"` (default), `"dominant"` or `"recessive"`; the dominant model recodes dosages as `min(G, 1)`, and the recessive model recodes as `max(G-1, 0)`

res.savefn	an RData or GDS file name, "" for no saving
res.compress	the compression method for the output file, it should be one of LZMA, LZMA_RA, ZIP, ZIP_RA and none
parallel	FALSE (serial processing), TRUE (multicore processing), a numeric value for the number of cores, or other value; parallel is passed to the argument cl in seqParallel, see seqParallel for more details
verbose	if TRUE, show information
verbose.maf	if TRUE, show summary of MAFs in units

Details

seqUnitFilterCond() in the SeqArray package can be used to restrict the variant sets units to a range of MAC and/or MAF.

For more details of the ACAT-V method, please refer to the ACAT paper [Liu et al. 2019] (see the reference section).

When geno.model="dominant", the dosages are recoded as min(G, 1) after allele flipping to the minor allele, testing whether having at least one copy of the minor allele is associated with the trait. When geno.model="recessive", dosages are recoded as max(G-1, 0), testing whether having two copies of the minor allele is associated. MAF/MAC filtering and the reported allele frequencies use the original (additive) genotype values regardless of the genetic model.

Value

Return a data.frame with the following components if not saving to a file:

chr	chromosome;
start	the starting position;
end	the ending position;
maxMAF	maximum of MAFs;
numvar	the number of variants in the window; if weight="Cauchy", it is the number of individual p-values in the Cauchy test
macmin	the minimum of MAC in the window;
macmed	the median of MAC in the window;
macmax	the maximum of MAC in the window;
summac	the sum of minor allele counts;
weight	the weighting strategy with the beta distribution parameters
n_single	the number of variants running for single variant tests;
n_collapse	the number of ultra rare variants for collapsing burden tests (<= collapse.mac);
pval	ACAT-V p-value;

Author(s)

Xiuwen Zheng

References

Liu Y., Chen S., Li Z., Morrison A.C., Boerwinkle E., Lin X. ACAT: A Fast and Powerful p Value Combination Method for Rare-Variant Analysis in Sequencing Studies. Am J Hum Genetics 104, 410-421 (2019).

See Also

seqAssocGLMM_ACAT_O, seqAssocGLMM_Burden, seqAssocGLMM_SKAT, seqUnitFilterCond

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, trait.type="binary")

# get a list of variant units for burden tests
units <- seqUnitSlidingWindows(gdsfile, win.size=500, win.shift=250)

a <- seqAssocGLMM_ACAT_V(gdsfile, glmm, units, maxMAF=c(0.1,0.01))
head(a)

# close the GDS file
seqClose(gdsfile)
```

---

seqAssocGLMM_Burden *Burden tests*

Description

Set-based burden tests using mixed models.

Usage

```
seqAssocGLMM_Burden(gdsfile, modobj, units, maxMAF=0.01, wbeta=AggrParamBeta,
    missing=0.05, ccimb.adj=TRUE, ER.mac=4.5, dsnode="",
    geno.model=c("additive", "dominant", "recessive"),
    res.savefn="", res.compress="ZIP", parallel=FALSE, verbose=TRUE,
    verbose.maf=FALSE)
```

Arguments

gdsfile	a SeqArray GDS filename, or a GDS object
modobj	an R object for SAIGE model parameters
units	a list of units of selected variants, with S3 class "SeqUnitListClass" defined in the SeqArray package
maxMAF	a numeric vector for MAF thresholds, e.g., c(0.01, 0.001) for rare variants, or c(0.5, 0.1, 0.01) for both common and rare variants
wbeta	weights for per-variant effect, using beta distribution dbeta() according to variant's MAF; a length-two vector, or a matrix with two rows for multiple beta parameters; by default, using beta(1,1) and beta(1,25) both
missing	missing threshold for variants (checking <= missing), NaN for no filter
ccimb.adj	whether adjusting for case-control imbalance or not
ER.mac	the maximum sum of MAC for the variants with efficient resampling p-values
dsnode	"" for automatically searching the GDS nodes "genotype" and "annotation/format/DS", or use a user-defined GDS node in the file
geno.model	a character string for the genetic model: "additive" (default), "dominant" or "recessive"; the dominant model recodes dosages as min(G, 1), and the recessive model recodes as max(G-1, 0)
res.savefn	an RData or GDS file name, "" for no saving
res.compress	the compression method for the output file, it should be one of LZMA, LZMA_RA, ZIP, ZIP_RA and none
parallel	FALSE (serial processing), TRUE (multicore processing), a numeric value for the number of cores, or other value; parallel is passed to the argument cl in seqParallel, see seqParallel for more details
verbose	if TRUE, show information
verbose.maf	if TRUE, show summary of MAFs in units

Details

seqUnitFilterCond() in the SeqArray package can be used to restrict the variant sets units to a range of MAC and/or MAF.

When geno.model="dominant", the dosages are recoded as min(G, 1) after allele flipping to the minor allele, testing whether having at least one copy of the minor allele is associated with the trait. When geno.model="recessive", dosages are recoded as max(G-1, 0), testing whether having two copies of the minor allele is associated. MAF/MAC filtering and the reported allele frequencies use the original (additive) genotype values regardless of the genetic model.

Value

Return a data.frame with the following components if not saving to a file:

chr	chromosome;
start	the starting position;

end	the ending position;
maxMAF	maximum of MAFs;
numvar	the number of variants in the window; if weight="Cauchy", it is the number of individual p-values in the Cauchy test
macmin	the minimum of MAC in the window;
macmed	the median of MAC in the window;
macmax	the maximum of MAC in the window;
summac	the sum of minor allele counts;
weight	the weighting strategy with the beta distribution parameters
beta	beta coefficient, log odds ratio if binary outcomes (alternative allele vs. reference allele);
SE	standard error for beta coefficient;
pval	final p-value (if the Saddlepoint approximation or efficient resampling is applied);
method	"Normal": pval is calculated by assuming asymptotic normality; "SPA": using the Saddlepoint approximation method; "ER": using the efficient resampling
p.norm	p-values based on asymptotic normality (could be 0 if it is too small, e.g., pnorm(-50) = 0 in R; used for checking only
converged	whether the SPA algorithm converges or not for p-values.

Author(s)

Xiuwen Zheng

See Also

seqAssocGLMM_ACAT_V, seqAssocGLMM_ACAT_O, seqAssocGLMM_SKAT, seqUnitFilterCond

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, trait.type="binary")

# get a list of variant units for burden tests
units <- seqUnitSlidingWindows(gdsfile, win.size=500, win.shift=250)

a <- seqAssocGLMM_Burden(gdsfile, glmm, units, maxMAF=c(0.1,0.01))
```

```
head(a)

# close the GDS file
seqClose(gdsfile)
```

`seqAssocGLMM_SKAT` *SKAT tests*

Description

Set-based SKAT tests using mixed models.

Usage

```
seqAssocGLMM_SKAT(gdsfile, modobj, units, maxMAF=0.01, wbeta=AggrParamBeta,
    missing=0.05, collapse.mac=10, collapse.method=c("max", "sum"),
    ccimb.adj=TRUE, ER.mac=4.5, dsnode="",
    geno.model=c("additive", "dominant", "recessive"),
    res.savefn="", res.compress="ZIP",
    parallel=FALSE, verbose=TRUE, verbose.maf=FALSE)
```

Arguments

`gdsfile`	a SeqArray GDS filename, or a GDS object
`modobj`	an R object for SAIGE model parameters
`units`	a list of units of selected variants, with S3 class `"SeqUnitListClass"` defined in the SeqArray package
`maxMAF`	a numeric vector for MAF thresholds, e.g., `c(0.01, 0.001)` for rare variants, or `c(0.5, 0.1, 0.01)` for both common and rare variants
`wbeta`	weights for per-variant effect, using beta distribution `dbeta()` according to variant's MAF; a length-two vector, or a matrix with two rows for multiple beta parameters; by default, using beta(1,1) and beta(1,25) both
`missing`	missing threshold for variants (checking `<= missing`), `NaN` for no filter
`collapse.mac`	the mac threshold for collapsing ultra rare variants
`collapse.method`	maximum of dosages of rare variants (`"max"`, by default), sum up rare genotypes (`"sum"`)
`ccimb.adj`	whether adjusting for case-control imbalance
`ER.mac`	the maximum sum of MAC for the variants with efficient resampling p-values
`dsnode`	"" for automatically searching the GDS nodes "genotype" and "annotation/format/DS", or use a user-defined GDS node in the file
`geno.model`	a character string for the genetic model: `"additive"` (default), `"dominant"` or `"recessive"`; the dominant model recodes dosages as `min(G, 1)`, and the recessive model recodes as `max(G-1, 0)`

res.savefn	an RData or GDS file name, "" for no saving
res.compress	the compression method for the output file, it should be one of LZMA, LZMA_RA, ZIP, ZIP_RA and none
parallel	FALSE (serial processing), TRUE (multicore processing), a numeric value for the number of cores, or other value; parallel is passed to the argument cl in seqParallel, see seqParallel for more details
verbose	if TRUE, show information
verbose.maf	if TRUE, show summary of MAFs in units

Details

seqUnitFilterCond() in the SeqArray package can be used to restrict the variant sets units to a range of MAC and/or MAF.

For more details of the SAIGE-GENE method, please refer to the SAIGE paper [Zhou et al. 2020] (see the reference section).

When geno.model="dominant", the dosages are recoded as min(G, 1) after allele flipping to the minor allele, testing whether having at least one copy of the minor allele is associated with the trait. When geno.model="recessive", dosages are recoded as max(G-1, 0), testing whether having two copies of the minor allele is associated. MAF/MAC filtering and the reported allele frequencies use the original (additive) genotype values regardless of the genetic model.

Value

Return a data.frame with the following components if not saving to a file:

chr	chromosome;
start	the starting position;
end	the ending position;
maxMAF	maximum of MAFs;
numvar	the number of variants in the window; if weight="Cauchy", it is the number of individual p-values in the Cauchy test
macmin	the minimum of MAC in the window;
macmed	the median of MAC in the window;
macmax	the maximum of MAC in the window;
summac	the sum of minor allele counts;
weight	the weighting strategy with the beta distribution parameters
n_collapse	the number of variants running for single variant tests;
g_ncol	the number of ultra rare variants for collapsing burden tests (<= collapse.mac);
g_minMAC	x
pval	SKAT p-value;

Author(s)

Xiuwen Zheng

References

Zhou W, Zhao Z, Nielsen JB, Fritsche LG, LeFaive J, Taliun SAG, Bi W, Gabrielsen ME, Daly MJ, Neale BM, Hveem K, Abecasis GR, Willer CJ, et al. Scalable generalized linear mixed model for region-based association tests in large biobanks and cohorts. Nat Genet. 2020; 52: 634-9.

See Also

seqAssocGLMM_ACAT_V, seqAssocGLMM_ACAT_O, seqAssocGLMM_Burden, seqUnitFilterCond

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

sp_grm_fn <- system.file("extdata", "grm1k_10k_sp_grm.rds", package="SAIGEgds")
sp_grm <- readRDS(sp_grm_fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, sp_grm,
    trait.type="binary")

# get a list of variant units for burden tests
units <- seqUnitSlidingWindows(gdsfile, win.size=50, win.shift=250)

a <- seqAssocGLMM_SKAT(gdsfile, glmm, units, maxMAF=c(0.1,0.01))
head(a)

# close the GDS file
seqClose(gdsfile)
```

---

seqAssocGLMM_SPA *P-value calculation*

Description

P-value calculations using variance approximation and an adjustment of Saddlepoint approximation in the mixed model framework.

Usage

```
seqAssocGLMM_SPA(gdsfile, modobj, maf=NaN, mac=10, missing=0.05, spa=TRUE,
    ER.mac=4.5, dsnode="",
    geno.model=c("additive", "dominant", "recessive"), geno.ploidy=2L,
```

```
    res.savefn="", res.compress="ZIP",
    parallel=FALSE, load.balancing=TRUE,
    verbose.pval=c(0, 5e-10, 5e-8, 5e-6, 5e-4, 1), verbose=TRUE)
seqAssocGLMM_GT(gt, modobj, spa=TRUE, ER.mac=4.5,
    geno.model=c("additive", "dominant", "recessive"), geno.ploidy=2L,
    verbose=TRUE)
```

Arguments

gdsfile	a SeqArray GDS filename, or a GDS object
modobj	an R object for SAIGE model parameters
maf	minor allele frequency threshold (checking >= maf), NaN for no filter
mac	minor allele count threshold (checking >= mac), NaN for no filter
missing	missing threshold for variants (checking <= missing), NaN for no filter
spa	TRUE for using the Saddlepoint approximation method for adjusting case-control imbalance
ER.mac	the maximum MAC for the variants with efficient resampling p-values
dsnode	"" for automatically searching the GDS nodes "genotype" and "annotation/format/DS", or use a user-defined GDS node in the file
geno.model	a character string for the genetic model: "additive" (default), "dominant" or "recessive"; the dominant model recodes dosages as min(G, 1) (having >= 1 copy of the minor allele), and the recessive model recodes as max(G-1, 0) (having 2 copies of the minor allele); only applicable when geno.ploidy > 0
geno.ploidy	specify the ploidy (2 by default); 0 or NA used for non-genotype data (e.g., CNV and dsnode should be specified for CNVs meanwhile)
res.savefn	an RData or GDS file name, "" for no saving
res.compress	the compression method for the output file, it should be one of ZIP, ZIP_RA, LZMA, LZMA_RA and none; see compression.gdsn for more details
parallel	FALSE (serial processing), TRUE (multicore processing), a numeric value for the number of cores, or other value; parallel is passed to the argument cl in seqParallel, see seqParallel for more details
load.balancing	load balancing for the calculation in parallel if TRUE
gt	a numeric genotype matrix (sample by variant) or a Matrix object, where rows correspond to the samples in the model and columns to variants
verbose.pval	NULL or a numeric vector for dividing p-values into the intervals and show the corresponding counts
verbose	if TRUE, show information

Details

seqAssocGLMM_SPA() reads genotype data from a GDS file and supports parallel computation. seqAssocGLMM_GT() takes an in-memory numeric matrix gt (sample by variant) directly, which is useful when genotypes are already loaded or generated outside of the GDS framework. The rows of gt must match the samples used in the null model.

For more details of SAIGE algorithm, please refer to the SAIGE paper [Zhou et al. 2018] (see the reference section).

When `geno.model="dominant"`, the dosages are recoded as `min(G, 1)` after allele flipping to the minor allele, testing whether having at least one copy of the minor allele is associated with the trait. When `geno.model="recessive"`, dosages are recoded as `max(G-1, 0)`, testing whether having two copies of the minor allele is associated. MAF/MAC filtering and the reported allele frequencies use the original (additive) genotype values regardless of the genetic model.

Value

Return a `data.frame` with the following components if not saving to a file:

`id`	variant ID in the GDS file;
`chr`	chromosome;
`pos`	position;
`rs.id`	the RS IDs if it is available in the GDS file as a node "annotation/id";
`ref`	the reference allele;
`alt`	the alternative allele;
`AF.alt`	allele frequency for the alternative allele; the minor allele frequency is `pmin(AF.alt, 1-AF.alt)`;
`mac`	minor allele count; the allele count for the alternative allele is `ifelse(AF.alt<=0.5, mac, 2*num-mac)`;
`num`	the number of samples with non-missing genotypes;
`beta`	beta coefficient, log odds ratio if binary outcomes (alternative allele vs. reference allele);
`SE`	standard error for beta coefficient;
`pval`	final p-value (if the Saddlepoint approximation or efficient resampling is applied);
`method`	`"Normal"`: `pval` is calculated by assuming asymptotic normality; `"SPA"`: using the Saddlepoint approximation method; `"ER"`: using the efficient resampling
`p.norm`	p-values based on asymptotic normality (could be 0 if it is too small, e.g., `pnorm(-50) = 0` in R; used for checking only
`converged`	whether the SPA algorithm converges or not for p-values.

Author(s)

Xiuwen Zheng

References

Zhou W, Nielsen JB, Fritsche LG, Dey R, Gabrielsen ME, Wolford BN, LeFaive J, VandeHaar P, Gagliano SA, Gifford A, Bastarache LA, Wei WQ, Denny JC, Lin M, Hveem K, Kang HM, Abecasis GR, Willer CJ, Lee S. Efficiently controlling for case-control imbalance and sample relatedness in large-scale genetic association studies. Nat Genet (2018). Sep;50(9):1335-1341.

See Also

seqFitNullGLMM_SPA, seqSAIGE_LoadPval

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, trait.type="binary")

# p-value calculation
a <- seqAssocGLMM_SPA(gdsfile, glmm, mac=10)
head(a)

# close the GDS file
seqClose(gdsfile)
```

---

seqFitLDpruning *Linkage disequilibrium pruning*

Description

Construct LD-pruned SNP sets for genetic relationship matrix (GRM).

Usage

```
seqFitLDpruning(gdsfile, sample.id=NULL, variant.id=NULL,
    ld.threshold=0.1, maf=0.01, missing.rate=0.005, autosome.only=TRUE,
    use.cateMAC=TRUE, num.marker=100L, num.total=100000L, save.gdsfn=NULL,
    seed=200L, parallel=FALSE, parallel.multi=NULL, verbose=TRUE)
```

Arguments

gdsfile	a SeqArray GDS file name, a GDS object for genotypes, or a character vector for a list of GDS file names when genotyeps are split by chromosomes
sample.id	NULL for all samples, or sample IDs in GRM
variant.id	a list of variant IDs, used to construct GRM
ld.threshold	the LD threshold; see snpgdsLDpruning()
maf	minor allele frequency threshold for genotypes in gdsfile (checking >= maf), if variant.id=NULL; NaN for no filter

missing.rate	threshold of missing rate (checking <= missing.rate), if variant.id=NULL; NaN for no filter
autosome.only	TRUE for using autosomes only
use.cateMAC	FALSE, to use a single global variance ratio; TRUE (equal to use.cateMAC=c(1.5, 2.5, 3.5, 4.5, 5.5, 10.5, 20.5)) for MAC categories (0, 1.5), [1.5, 2.5), ... [10.5, 20.5) and [20.5, Inf); or a numeric vector (strictly increasing) for unique cut points
num.marker	the number of SNPs used to calculate the variance ratio in each MAC category
num.total	the total number of LD-pruned variants excluding ultra rare variants; if the number of variants selected by the process of LD pruning is larger than num.total, the random set is used
save.gdsfn	if a file name is specified, construct a GDS genotype file to include all selected variants
parallel	FALSE (serial processing), TRUE (multicore processing), a numeric value for the number of cores, or other value; parallel is passed to the argument cl in seqParallel, see seqParallel for more details
parallel.multi	only applicable when there are multiple input files in gdsfile and save.gdsfn is used; multiple GDS files can be extracted to create a sub GDS file in parallel via parallel.multi; if NULL, use parallel instead; parallel.multi could have less CPU cores than parallel to reduce memory usage
seed	an integer passed to set.seed() as a seed for random numbers, or NULL for no action
verbose	if TRUE, show information

Details

This function calls snpgdsLDpruning in the SNPRelate package to perform linkage disequilibrium (LD) pruning. When use.cateMAC is not FALSE, the ultra rare variants will be selected according to the MAC categories, which could be used in the null model fitting.

Value

Returns variant IDs or a list of variant IDs for multiple input GDS files.

Author(s)

Xiuwen Zheng

See Also

seqFitNullGLMM_SPA, seqAssocGLMM_SPA, snpgdsLDpruning

Examples

```
library(SeqArray)
library(SAIGEgds)

# open a GDS file
gds_fn <- seqExampleFileName("KG_Phase1")

seqFitLDpruning(gds_fn, save.gdsfn="grm.gds")

# delete the temporary file
unlink("grm.gds", force=TRUE)
```

`seqFitNullGLMM_SPA` *Fit the null model for the SAIGE mixed model framework*

Description

Fit the null model in the mixed model framework with an optional genetic relationship matrix (GRM).

Usage

```
seqFitNullGLMM_SPA(formula, data, gdsfile=NULL, grm.mat=NULL,
    trait.type=c("binary", "quantitative"), sample.col="sample.id", maf=0.01,
    missing.rate=0.01, max.num.snp=1000000L, variant.id=NULL,
    variant.id.varratio=NULL, nsnp.sub.random=2000L, rel.cutoff=0.125,
    inv.norm=c("residuals", "quant", "none"), use.cateMAC=FALSE,
    cateMAC.inc.maf=TRUE, cateMAC.simu=TRUE, use.offset=FALSE, X.transform=TRUE,
    tol=0.02, maxiter=20L, nrun=30L, tolPCG=1e-5, maxiterPCG=500L,
    num.marker=30L, tau.init=c(0,0), traceCVcutoff=0.0025, ratioCVcutoff=0.001,
    geno.sparse=TRUE, use.gpu=FALSE, save.packed.geno=FALSE, num.thread=1L,
    model.savefn="", seed=200L, fork.loading, parallel.loading=FALSE,
    verbose=TRUE)
seqRefitNullGLMM(formula, data, model=NULL, sample.col="sample.id",
    inv.norm=c("residuals", "quant", "none"), use.offset=FALSE,
    X.transform=TRUE, tau.update=FALSE, recalcVR=FALSE,
    tol=0.02, maxiter=20L, nrun=30L, tolPCG=1e-5, maxiterPCG=500L,
    num.marker=30L, traceCVcutoff=0.0025, ratioCVcutoff=0.001,
    num.thread=1L, seed=200L, use.gpu=FALSE, verbose=TRUE)
```

Arguments

`formula`	an object of class `formula` (or one that can be coerced to that class), e.g., `y ~ x1 + x2`, see `lm`
`data`	a data frame for the formulas
`gdsfile`	a SeqArray GDS filename or a GDS object for genotypes used in the GRM calculation; see details

grm.mat	NULL, a user-defined GRM or an object of 'snpgdsGRMClass' returned from SNPRelate::snpgdsGRM(): a dense numeric matrix or a sparse symmetric matrix 'sparseMatrix' defined in Matrix; colnames and rownames should be sample IDs; or TRUE, call seqFitSparseGRM() to get a sparse GRM; see details
trait.type	"binary" for binary outcomes, "quantitative" for continuous outcomes
sample.col	the column name of sample IDs corresponding to the GDS file
maf	minor allele frequency threshold for genotypes in gdsfile (checking >= maf), if variant.id=NULL; NaN for no filter
missing.rate	threshold of missing rate (checking <= missing.rate), if variant.id=NULL; NaN for no filter
max.num.snp	the maximum number of SNPs used, or -1 for no limit
variant.id	a list of variant IDs, considered to be used in GRM; or NULL, all variants in the GDS file to be the candidate variants
variant.id.varratio	a list of variant IDs, to be used in the variance ratio estimation; or NULL to use the candidate variants according to variant.id
nsnp.sub.random	used in seqFitSparseGRM() when grm.mat=TRUE: the number of SNP markers randomly selected from the candidate variants for the initial scan of relatedness
rel.cutoff	relatedness threshold for treating two individuals as unrelated (check >= rel.cutoff), NaN or -Inf for no threshold; only applicable when grm.mat=TRUE
use.cateMAC	FALSE, to use a single global variance ratio; TRUE (equal to use.cateMAC=c(1.5, 2.5, 3.5, 4.5, 5.5, 10.5, 20.5)) for MAC categories (0, 1.5), [1.5, 2.5), ... [10.5, 20.5) and [20.5, Inf); or a numeric vector (strictly increasing) for unique cut points
cateMAC.inc.maf	TRUE to include a MAC cut point according to MAF=maf, or a numeric vector for MAF; only applicable if use.cateMAC is not FALSE
cateMAC.simu	TRUE for using simulated independent genotypes under Hardy-Weinberg equilibrium if there are no enough SNP markers in the MAC category; only applicable if use.cateMAC is not FALSE
inv.norm	"residuals" (by default), perform the inverse normal transformation on the residuals after fitting the fixed effects; "quant", perform the inverse normal transformation on the outcome variable directly (the same behavior as the SAIGE package); "none", no transformation. If inv.norm=TRUE, perform inv.norm="residuals"; if inv.norm=FALSE, it is as the same as inv.norm="none"; See the reference for more discussion [Sofer et al., 2019]
use.offset	if TRUE, use the estimated residual offset to avoid calculating each fixed effect coefficient
X.transform	if TRUE, perform QR decomposition on the design matrix
tol	overall tolerance for model fitting
maxiter	the maximum number of iterations for model fitting; maxiter=0 for using a fixed tau parameter in tau.init
nrun	the number of random vectors in the trace estimation

tolPCG	tolerance of PCG iterations
maxiterPCG	the maximum number of PCG iterations
num.marker	the number of SNPs used to calculate the variance ratio
tau.init	a 2-length numeric vector, the initial values for variance components, tau; for binary traits, the first element is always be set to 1. if tau.init is not specified, the second element will be 0.5 for binary traits
traceCVcutoff	the threshold for coefficient of variation (CV) for the trace estimator, and the number of runs for trace estimation will be increased until the CV is below the threshold
ratioCVcutoff	the threshold for coefficient of variation (CV) for estimating the variance ratio, and the number of randomly selected markers will be increased until the CV is below the threshold
geno.sparse	if TRUE (by default), store the sparse structure for genotypes; otherwise, save genotypes in a 2-bit dense matrix; see details
save.packed.geno	if TRUE, save packed genotypes of GRM in the output model object; it can be used for refitting the null model with different fixed effects
num.thread	the number of threads, 1 by default
model.savefn	the filename of model output, R data file '.rda', '.RData', or '.rds'
seed	an integer passed to set.seed() as a seed for random numbers, or NULL for no action
fork.loading	deprecated, use parallel.loading instead
parallel.loading	load genotypes via parallel or not; multiple processes can reduce loading time of genotypes, but may increase the memory usage
model	a fitted null model object of class "ClassSAIGE_NullModel" created by seqFitNullGLMM_SPA() with save.packed.geno=TRUE
tau.update	if TRUE, re-estimate the variance components tau; otherwise use the tau values from the input model
recalcVR	if TRUE, recalculate the variance ratio; otherwise use the variance ratio from the input model
use.gpu	if TRUE, attempt to use OpenCL GPU acceleration for the GRM cross-product computation; falls back to CPU if no compatible GPU or OpenCL runtime is found
verbose	if TRUE, show information

Details

seqFitNullGLMM_SPA() fits the null model from scratch using genotype data from a GDS file and/or a genetic relationship matrix (GRM). seqRefitNullGLMM() refits the null model using packed genotypes saved from a previous call to seqFitNullGLMM_SPA() (with save.packed.geno=TRUE), allowing different fixed-effect covariates without reloading genotypes from the GDS file. By default, the variance components (tau) are kept from the input model (tau.update=FALSE) and the variance ratio is recalculated (recalcVR=TRUE).

Utilizing the sparse structure of genotypes could significantly improve the computational efficiency of model fitting, but it also increases the memory usage. For more details of SAIGE algorithm, please refer to the SAIGE paper [Zhou et al., 2018] (see the reference section).

Value

Returns a list with the following components:

`coefficients`	the beta coefficients for fixed effects;
`tau`	a numeric vector of variance components 'Sigma_E' and 'Sigma_G';
`linear.predictors`	the linear fit on link scale;
`fitted.values`	fitted values from objects returned by modeling functions using `glm.fit`;
`residuals`	residuals;
`converged`	whether the model is fitted or not;
`obj.noK`	internal use, returned object from the SPAtest package;
`Sigma_inv`	optional inverse covariance matrix when a GRM is used;
`chol_inv_X_Sigma`	optional projection matrix component when a GRM is used;
`var.ratio`	a data.frame with columns 'id' (variant.id), 'maf' (minor allele frequency), 'mac' (minor allele count), 'var1' (the variance of score statistic), 'var2' (a variance estimate without accounting for estimated random effects) and 'ratio' (var1/var2, estimated variance ratio for variance approximation);
`trait.type`	either "binary" or "quantitative";
`sample.id`	the sample IDs used in the model fitting;
`variant.id`	the variant IDs used in the model fitting.

Author(s)

Xiuwen Zheng

References

Zhou W, Nielsen JB, Fritsche LG, Dey R, Gabrielsen ME, Wolford BN, LeFaive J, VandeHaar P, Gagliano SA, Gifford A, Bastarache LA, Wei WQ, Denny JC, Lin M, Hveem K, Kang HM, Abecasis GR, Willer CJ, Lee S. Efficiently controlling for case-control imbalance and sample relatedness in large-scale genetic association studies. Nat Genet (2018). Sep;50(9):1335-1341.

Zhou W, Zhao Z, Nielsen JB, Fritsche LG, LeFaive J, Taliun SAG, Bi W, Gabrielsen ME, Daly MJ, Neale BM, Hveem K, Abecasis GR, Willer CJ, et al. Scalable generalized linear mixed model for region-based association tests in large biobanks and cohorts. Nat Genet. 2020; 52: 634-9.

See Also

`seqAssocGLMM_SPA`, `seqFitSparseGRM`

Examples

```
# open a GDS file
fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(fn)

# load phenotype
phenofn <- system.file("extdata", "pheno.txt.gz", package="SAIGEgds")
pheno <- read.table(phenofn, header=TRUE, as.is=TRUE)
head(pheno)

# fit the null model
glmm <- seqFitNullGLMM_SPA(y ~ x1 + x2, pheno, gdsfile, trait.type="binary")
glmm

# close the GDS file
seqClose(gdsfile)
```

---

seqFitSparseGRM *Sparse & dense genetic relationship matrix*

Description

Construct sparse and dense genetic relationship matrix (GRM).

Usage

```
seqFitSparseGRM(gdsfile, sample.id=NULL, variant.id=NULL, nsnp.sub.random=2000L,
    rel.cutoff=0.125, maf=0.01, missing.rate=0.005, num.thread=1L,
    use.gpu=FALSE, return.ID=FALSE, seed=200L, verbose=TRUE)
seqFitDenseGRM(gdsfile, sample.id=NULL, variant.id=NULL, maf=0.01,
    missing.rate=0.005, num.thread=1L, use.gpu=FALSE, use.double=TRUE,
    return.ID=FALSE, verbose=TRUE)
```

Arguments

gdsfile	a SeqArray GDS file name or a GDS object for genotypes used in the GRM calculation; see details
sample.id	NULL for all samples, or sample IDs in GRM
variant.id	candidate variant IDs used for constructing GRM; or NULL for using all available candidate variants
nsnp.sub.random	the number of SNP markers randomly selected from the candidate variants for the initial scan of relatedness; if nsnp.sub.random=0, use all candidate SNPs
rel.cutoff	relatedness threshold for treating two individuals as unrelated (check >= rel.cutoff); NaN or -Inf for no threshold
maf	minor allele frequency threshold for genotypes in gdsfile (checking >= maf), if variant.id=NULL; NaN for no filter

missing.rate	threshold of missing rate (checking <= missing.rate), if variant.id=NULL; NaN for no filter
num.thread	the number of threads, 1 by default
use.gpu	if TRUE, attempt to use OpenCL GPU acceleration for GRM computation; falls back to CPU if no GPU is available
use.double	TRUE for using 64-bit double precision to calculate GRM; otherwise, to use 32-bit single precision
return.ID	if TRUE, return the IDs for samples, the full variant set and the subset of variants
seed	an integer passed to set.seed() as a seed for random numbers, or NULL for no action
verbose	if TRUE, show information

Details

The genetic relationship matrix (GRM) is defined as $grm_ij = avg_l [(g_il - 2*p_l)*(g_jl - 2*p_l) / 2*p_l*(1 - p_l)]$ for individuals i,j and locus l, where g_il is 0, 1 or 2, and p_l is the allele frequency at locus l. The missing genotypes are dropped from the calculation.

Value

If return.ID=TRUE, returns a list with sample.id for sample IDs, variant.id for the full set of variants, variant.sub.id for the subset of variants, and the GRM matrix. Otherwise, it returns a sparse or dense symmetric matrix for GRM, with sample IDs in colnames() and rownames().

Author(s)

Xiuwen Zheng

See Also

seqFitNullGLMM_SPA, seqFitLDpruning

Examples

```
library(Matrix)

# open a GDS file
gds_fn <- system.file("extdata", "grm1k_10k_snp.gds", package="SAIGEgds")
gdsfile <- seqOpen(gds_fn)

seqSetFilter(gdsfile, variant.sel=1:100)
m <- seqFitSparseGRM(gdsfile, rel.cutoff=0.125)
is(m)
nnzero(m)                 # num of non-zero
nnzero(m) / prod(dim(m))  # percentage of non-zero

m <- seqFitDenseGRM(gdsfile)
str(m)
```

```
# close the GDS file
seqClose(gdsfile)
```

---

seqGetGenoByGeno *Genotype-by-Genotype Interaction Matrix*

Description

Calculate the matrix for genotype-by-genotype interaction terms between two sets of variants, excluding interaction columns without variation.

Usage

```
seqGetGenoByGeno(gt1, gt2, use_colnm=TRUE)
```

Arguments

gt1	a numeric matrix or Matrix object (samples by variants) for the first set of genotypes
gt2	a numeric matrix or Matrix object (samples by variants) for the second set of genotypes
use_colnm	if TRUE, set column names in the output matrix using the column names of gt1 and gt2; if column names are missing, default names "a1", "a2", ... and "b1", "b2", ... are used

Details

The function computes element-wise products for all pairs of columns (one from gt1, one from gt2) to form interaction terms. Interaction columns with no variation (i.e., all zeros) are excluded from the result. The computation is performed using sparse matrix operations for efficiency, and the inputs are swapped internally if gt1 has more columns than gt2.

Value

A sparse matrix (dgCMatrix) where each column represents the interaction between a variant in gt1 and a variant in gt2. Column names are in the format "name1_x_name2" when use_colnm=TRUE.

Author(s)

Xiuwen Zheng

See Also

seqAssocGLMM_SPA

Examples

```
library(Matrix)

# dense matrix input
g1 <- matrix(c(0,1,2,0,1,0), nrow=3)
g2 <- matrix(c(1,0,1,2,1,0), nrow=3)
colnames(g1) <- c("snp1", "snp2")
colnames(g2) <- c("snp3", "snp4")

m <- seqGetGenoByGeno(g1, g2)
m

# sparse matrix input
g1s <- as(g1, "dgCMatrix")
g2s <- as(g2, "dgCMatrix")

m2 <- seqGetGenoByGeno(g1s, g2s)
m2
```

`seqSAIGE_LoadPval` *Load the association results*

Description

Load the association results from an RData, RDS or GDS file.

Usage

```
seqSAIGE_LoadPval(fn, varnm=NULL, index=NULL, verbose=TRUE)
```

Arguments

`fn`	RData, RDS or GDS file names, merging datasets if multiple files
`varnm`	NULL, or a character vector to include the column names; e.g., `c("chr", "position", "rs.id", "ref", "alt", "pval")`
`index`	NULL, or a logical/numeric vector for a set of rows
`verbose`	if `TRUE`, show information

Value

Return a `data.frame` including p-values.

Author(s)

Xiuwen Zheng

See Also

`seqFitNullGLMM_SPA`, `seqAssocGLMM_SPA`

Examples

```
(fn <- system.file("unitTests", "saige_pval.rds", package="SAIGEgds"))
pval <- seqSAIGE_LoadPval(fn)

names(pval)
#  [1] "id"         "chr"        "pos"        "rs.id"      "ref"
#  [6] "alt"        "AF.alt"     "AC.alt"     "num"        "beta"
# [11] "SE"         "pval"       "pval.noadj" "converged"

head(pval)
```

Index

∗ **Cauchy**
pACAT, 5
∗ **GDS**
glmmHeritability, 4
SAIGEgds-package, 2
seqAssocGLMM_ACAT_O, 6
seqAssocGLMM_ACAT_V, 9
seqAssocGLMM_Burden, 11
seqAssocGLMM_SKAT, 14
seqAssocGLMM_SPA, 16
seqFitLDpruning, 19
seqFitNullGLMM_SPA, 21
seqFitSparseGRM, 25
seqGetGenoByGeno, 27
seqSAIGE_LoadPval, 28
∗ **association**
glmmHeritability, 4
pACAT, 5
SAIGEgds-package, 2
seqAssocGLMM_ACAT_O, 6
seqAssocGLMM_ACAT_V, 9
seqAssocGLMM_Burden, 11
seqAssocGLMM_SKAT, 14
seqAssocGLMM_SPA, 16
seqFitLDpruning, 19
seqFitNullGLMM_SPA, 21
seqFitSparseGRM, 25
seqSAIGE_LoadPval, 28
∗ **genetics**
glmmHeritability, 4
SAIGEgds-package, 2
seqAssocGLMM_ACAT_O, 6
seqAssocGLMM_ACAT_V, 9
seqAssocGLMM_Burden, 11
seqAssocGLMM_SKAT, 14
seqAssocGLMM_SPA, 16
seqFitLDpruning, 19
seqFitNullGLMM_SPA, 21
seqFitSparseGRM, 25
seqGetGenoByGeno, 27
seqSAIGE_LoadPval, 28

compression.gdsn, *17*

glmmHeritability, 4

lm, *21*

pACAT, 5
pACAT2 *(*pACAT*)*, 5

SAIGEgds *(*SAIGEgds-package*)*, 2
SAIGEgds-package, 2
seqAssocGLMM_ACAT_O, 6, *11*, *13*, *16*
seqAssocGLMM_ACAT_O_GT *(*seqAssocGLMM_ACAT_O*)*, 6
seqAssocGLMM_ACAT_V, *8*, 9, *13*, *16*
seqAssocGLMM_Burden, *8*, *11*, 11, *16*
seqAssocGLMM_GT *(*seqAssocGLMM_SPA*)*, 16
seqAssocGLMM_SKAT, *8*, *11*, *13*, 14
seqAssocGLMM_SPA, *5*, 16, *20*, *24*, *27, 28*
seqFitDenseGRM *(*seqFitSparseGRM*)*, 25
seqFitLDpruning, 19, *26*
seqFitNullGLMM_SPA, *4, 5*, *19, 20*, 21, *26*, *28*
seqFitSparseGRM, *24*, 25
seqGetGenoByGeno, 27
seqParallel, *7*, *10*, *12*, *15*, *17*, *20*
seqRefitNullGLMM *(*seqFitNullGLMM_SPA*)*, 21
seqSAIGE_LoadPval, *19*, 28
seqUnitFilterCond, *8*, *11*, *13*, *16*
snpgdsLDpruning, *20*