

Package ‘GENESIS’

April 11, 2018

Type Package

Title GENetic ESTimation and Inference in Structured samples
(GENESIS): Statistical methods for analyzing genetic data from
samples with population structure and/or relatedness

Version 2.8.1

Date 2018-01-18

Author Matthew P. Conomos, Timothy Thornton, Stephanie M. Gogarten, Lisa Brown

Maintainer Matthew P. Conomos <mconomos@uw.edu>, Stephanie M. Gogarten <sdmorris@uw.edu>

Description The GENESIS package provides methodology for estimating, inferring, and accounting for population and pedigree structure in genetic analyses. The current implementation provides functions to perform PC-AiR (Conomos et al., 2015, Gen Epi) and PC-Relate (Conomos et al., 2016, AJHG). PC-AiR performs a Principal Components Analysis on genome-wide SNP data for the detection of population structure in a sample that may contain known or cryptic relatedness. Unlike standard PCA, PC-AiR accounts for relatedness in the sample to provide accurate ancestry inference that is not confounded by family structure. PC-Relate uses ancestry representative principal components to adjust for population structure/ancestry and accurately estimate measures of recent genetic relatedness such as kinship coefficients, IBD sharing probabilities, and inbreeding coefficients. Additionally, functions are provided to perform efficient variance component estimation and mixed model association testing for both quantitative and binary phenotypes.

License GPL-3

Depends

Imports Biobase, BiocGenerics, GWASTools, gdsfmt, GenomicRanges,
graph, IRanges, S4Vectors, SeqArray, SeqVarTools, graphics,
grDevices, methods, stats, utils

Suggests CompQuadForm, logistf, survey, SNPRelate, GWASdata, RUnit,
knitr

VignetteBuilder knitr

biocViews SNP, GeneticVariability, Genetics, StatisticalMethod,
DimensionReduction, PrincipalComponent, GenomeWideAssociation,
QualityControl, BiocViews

NeedsCompilation no

R topics documented:

GENESIS-package	2
admixmapMM	3
assocTestMM	5
assocTestSeq	9
assocTestSeqWindow	12
fitNullMM	16
fitNullReg	20
HapMap_ASW_MXL_KINGmat	21
king2mat	22
pcair	23
pcairPartition	27
pccrelate	29
pccrelateMakeGRM	32
pccrelateReadInbreed	33
pccrelateReadKinship	34
plot.pcair	35
varCompCI	37
Index	39

GENESIS-package	<i>GENetic Estimation and Inference in Structured samples (GENESIS): Statistical methods for analyzing genetic data from samples with pop- ulation structure and/or relatedness</i>
-----------------	---

Description

The GENESIS package provides methodology for estimating, inferring, and accounting for population and pedigree structure in genetic analyses. The current implementation performs PC-AiR (Conomos et al., 2015, Gen Epi) and PC-Relate (Conomos et al., 2016, AJHG). PC-AiR performs a Principal Components Analysis on genome-wide SNP data for the detection of population structure in a sample that may contain known or cryptic relatedness. Unlike standard PCA, PC-AiR accounts for relatedness in the sample to provide accurate ancestry inference that is not confounded by family structure. PC-Relate uses ancestry representative principal components to adjust for population structure/ancestry and accurately estimate measures of recent genetic relatedness such as kinship coefficients, IBD sharing probabilities, and inbreeding coefficients. Additionally, functions are provided to perform efficient variance component estimation and mixed model association testing for both quantitative and binary phenotypes.

Details

Package:	GENESIS
Type:	Package
Version:	2.1.7
Date:	2016-4-1
License:	GPL-3
Depends:	GWASTools
Suggests:	gdsfmt, SNPRelate, RUnit, BiocGenerics, knitr
VignetteBuilder:	knitr
biocViews:	SNP, GeneticVariability, Genetics, StatisticalMethod, DimensionReduction, PrincipalComponent, Gen

The PC-AiR analysis is performed using the `pcair` function, which takes genotype data and pairwise measures of kinship and ancestry divergence as input and returns PC-AiR PCs as the output. The function `pcairPartition` is called within `pcair` and uses the PC-AiR algorithm to partition the sample into an ancestry representative ‘unrelated subset’ and ‘related subset’. The function `plot.pcair` can be used to plot pairs of PCs from a class ‘pcair’ object returned by the function `pcair`. The function `king2mat` can be used to convert output text files from the KING software (Manichaikul et al., 2010) into an R matrix of pairwise kinship coefficient estimates in a format that can be used by the functions `pcair` and `pcairPartition`. The PC-Relate analysis is performed using the `pcrelate` function, which takes genotype data and PCs from PC-AiR and returns estimates of kinship coefficients, IBD sharing probabilities, and inbreeding coefficients. The functions `pcrelateReadKinship`, `pcrelateReadInbreed`, and `pcrelateMakeGRM` provide utilities for reading and making tables or matrices of the PC-Relate output. There are two functions required to perform SNP genotype association testing with mixed models. First, `fitNullMM` is called to fit the null model (i.e. no SNP genotype term) including fixed effects covariates, such as PC-AiR PCs, and random effects specified by their covariance structures, such as a kinship matrix created from PC-Relate output using `pcrelateMakeGRM`. The function `fitNullMM` uses AIREML to estimate variance components for the random effects, and the function `varCompCI` can be used to find confidence intervals on the estimates as well as the proportion of total variability they explain; this allows for heritability estimation. Second, `assocTestMM` is called with the null model output and the genotype data to perform either Wald or score based association tests.

Author(s)

Matthew P. Conomos and Timothy Thornton

Maintainer: Matthew P. Conomos <mconomos@uw.edu>

References

Conomos M.P., Reiner A.P., Weir B.S., & Thornton T.A. (2016). Model-free Estimation of Recent Genetic Relatedness. *American Journal of Human Genetics*, 98(1), 127-148.

Conomos M.P., Miller M., & Thornton T. (2015). Robust Inference of Population Structure for Ancestry Prediction and Correction of Stratification in the Presence of Relatedness. *Genetic Epidemiology*, 39(4), 276-293.

Gogarten, S. M., Bhangale, T., Conomos, M. P., Laurie, C. A., McHugh, C. P., Painter, I., ... & Laurie, C. C. (2012). GWASTools: an R/Bioconductor package for quality control and analysis of Genome-Wide Association Studies. *Bioinformatics*, 28(24), 3329-3331.

Manichaikul, A., Mychaleckyj, J.C., Rich, S.S., Daly, K., Sale, M., & Chen, W.M. (2010). Robust relationship inference in genome-wide association studies. *Bioinformatics*, 26(22), 2867-2873.

admixMapMM

admixMapMM

Description

Run admixture analyses

Usage

```
admixMapMM(admixDataList, nullMMobj, snp.include = NULL,
           chromosome = NULL, snp.block.size = 5000,
           verbose = TRUE)
```

Arguments

<code>admixDataList</code>	named list of GenotypeData objects for each ancestry
<code>nullMMobj</code>	A null model object returned by <code>fitNullMM</code> .
<code>snp.include</code>	A vector of SNP IDs to include in the analysis. If NULL, see <code>chromosome</code> for further details.
<code>chromosome</code>	A vector of integers specifying which chromosomes to analyze. This parameter is only considered when <code>snp.include</code> is NULL; if <code>chromosome</code> is also NULL, then all SNPs are included.
<code>snp.block.size</code>	The number of SNPs to read-in/analyze at once. The default value is 5000.
<code>verbose</code>	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Details

`admixDataList` should have one value for each ancestry. See the example for how one might set up this object. List names will propagate to the output file.

Value

data frame with admixture mapping results

Author(s)

Matthew P. Conomos, Lisa Brown

References

Brown, L.A. et al. (2017). Admixture Mapping Identifies an Amerindian Ancestry Locus Associated with Albuminuria in Hispanics in the United States. *J Am Soc Nephrol.* 28(7):2211-2220.

See Also

[GenotypeData](#), [fitNullMM](#), [assocTestMM](#)

Examples

```
library(GWASTools)
library(gdsfmt)

# create file with multiple ancestries
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
tmpfile <- tempfile()
file.copy(gdsfile, tmpfile)
gds <- openfn.gds(tmpfile, readonly=FALSE)
nsnp <- objdesp.gdsn(index.gdsn(gds, "snp.id"))$dim
nsamp <- objdesp.gdsn(index.gdsn(gds, "sample.id"))$dim
dosage_eur <- sample(0:2, nsnp*nsamp, replace=TRUE)
dosage_afr <- ifelse(dosage_eur == 2, 0, sample(0:1, nsnp*nsamp, replace=TRUE))
dosage_amer <- 2 - dosage_eur - dosage_afr
add.gdsn(gds, "dosage_eur", matrix(dosage_eur, nrow=nsamp, ncol=nsnp))
add.gdsn(gds, "dosage_afr", matrix(dosage_afr, nrow=nsamp, ncol=nsnp))
add.gdsn(gds, "dosage_amer", matrix(dosage_amer, nrow=nsamp, ncol=nsnp))
closefn.gds(gds)
```

```

# read GRM
pcrfile <- system.file("extdata", "HapMap_ASW_MXL_pcrelate.gds", package="GENESIS")
pcr <- openfn.gds(pcrfile)
mypcrel <- pcrelateMakeGRM(pcr)
closefn.gds(pcr)

# generate a phenotype
set.seed(4)
pheno <- rnorm(nsamp, mean = 0, sd = 1)
covar <- sample(0:1, nsamp, replace=TRUE)

# make ScanAnnotationDataFrame
scanAnnot <- ScanAnnotationDataFrame(data.frame(scanID = rownames(mypcrel),
      covar, pheno, stringsAsFactors=FALSE))

# read in GDS data
gds <- openfn.gds(tmpfile)
genoDataList <- list()
for (anc in c("eur", "afr", "amer")){
  gdsr <- GdsGenotypeReader(gds, genotypeVar=paste0("dosage_", anc))
  genoDataList[[anc]] <- GenotypeData(gdsr, scanAnnot=scanAnnot)
}

# fit the null mixed model
nullmod <- fitNullMM(scanData = scanAnnot, outcome = "pheno", covars = "covar", covMatList = mypcrel)

# run the association test
myassoc <- admixMapMM(genoDataList, nullMMobj = nullmod)

close(genoDataList[[1]])
unlink(tmpfile)

```

 assocTestMM

SNP Genotype Association Testing with Mixed Models

Description

assocTestMM performs SNP genotype association tests using the null model fit with [fitNullMM](#).

Usage

```

assocTestMM(genoData, nullMMobj, test = "Wald", snp.include = NULL,
  chromosome = NULL, impute.geno = TRUE, snp.block.size = 5000,
  ivars = NULL, ivar.return.betaCov = FALSE, verbose = TRUE)

```

Arguments

genoData An object of class `GenotypeData` from the package `GWASTools` containing the genotype data for SNPs and samples to be used for the analysis. This object can easily be created from a matrix of SNP genotype data, PLINK files, or GDS files. Alternatively, this could be an object of class `SeqVarData` from the package `SeqVarTools` containing the genotype data for the sequencing variants and samples to be used for the analysis.

<code>nullMMobj</code>	A null model object returned by <code>fitNullMM</code> .
<code>test</code>	A character string specifying the type of test to be performed. The possibilities are "Wald" (default) or "Score"; only "Score" can be used when the family of the null model fit with <code>fitNullMM</code> is not gaussian.
<code>snp.include</code>	A vector of SNP IDs to include in the analysis. If NULL, see <code>chromosome</code> for further details.
<code>chromosome</code>	A vector of integers specifying which chromosomes to analyze. This parameter is only considered when <code>snp.include</code> is NULL; if <code>chromosome</code> is also NULL, then all SNPs are included.
<code>impute.geno</code>	A logical indicator of whether sporadic missing genotype values should be mean imputed. The default is TRUE. See 'Details' for further information.
<code>snp.block.size</code>	The number of SNPs to read-in/analyze at once. The default value is 5000. See 'Details' for further information regarding how this parameter works when <code>impute.geno</code> is FALSE.
<code>ivars</code>	A vector of character strings specifying the names of the variables for which a genotype interaction term should be included. If NULL (default) no genotype interactions are included. See 'Details' for further information.
<code>ivar.return.betaCov</code>	Logical indicator of whether the estimated covariance matrix of the effect size estimates (betas) for the genotype and genotype interaction terms should be returned; the default is FALSE.
<code>verbose</code>	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Details

When `impute.geno` is TRUE, sporadic missing genotype values are mean imputed using the minor allele frequency (MAF) calculated on all other samples at that SNP. When `impute.geno` is FALSE, samples with missing values for all of the SNP genotypes in the current SNP block are removed from the analysis for the block; this may significantly slow down computation time because many pre-computed matrices need to be re-computed each time the sample set changes. Also note: when `impute.geno` is FALSE, sporadic missingness for a sample inside of a SNP block will lead to an error.

The input `ivars` can be used to perform GxE tests. Multiple interaction variables may be specified, but all interaction variables specified must have been included as covariates in fitting the null model with `fitNullMM`. When performing GxE analyses, `assocTestMM` will report two tests: (1) the joint test of all genotype interaction terms in the model (this is the test for any genotype interaction effect), and (2) the joint test of the genotype term along with all of the genotype interaction terms (this is the test for any genetic effect). Individual genotype interaction terms can be tested by creating Wald test statistics from the reported effect size estimates and their standard errors (Note: when `ivars` contains a single continuous or binary covariate, this test is the same as the test for any genotype interaction effect mentioned above). In order to test more complex hypotheses regarding subsets of multiple genotype interaction terms, `ivar.return.betaCov` can be used to retrieve the estimated covariance matrix of the effect size estimates.

Value

A data.frame where each row refers to a different SNP with the columns:

<code>snpID</code>	The SNP ID
<code>chr</code>	The numeric chromosome value

n The number of samples used to analyze the SNP
 MAF The estimated minor allele frequency
 minor.allele Either "A" or "B" indicating which allele is the minor allele

If test is "Score":

Score The value of the score function
 Var The variance of the score function
 Score.Stat The score chi-squared test statistic
 Score.pval The score p-value

If test is "Wald" and ivars is NULL:

Est The effect size estimate for each additional copy of the "A" allele
 SE The estimated standard error of the effect size estimate
 Wald.Stat The Wald chi-squared test statistic
 Wald.pval The Wald p-value

If test is "Wald" and ivars is not NULL:

Est.G The effect size estimate for the genotype term
 Est.G:ivar The effect size estimate for the genotype*ivar interaction term. There will be as many of these terms as there are interaction variables, and "ivar" will be replaced with the variable name.
 SE.G The estimated standard error of the genotype term effect size estimate
 SE.G:ivar The estimated standard error of the genotype*ivar effect size estimate. There will be as many of these terms as there are interaction variables, and "ivar" will be replaced with the variable name.
 GxE.Stat The Wald chi-squared test statistic for the test of all genotype interaction terms. When there is only one genotype interaction term, this is the test statistic for that term.
 GxE.pval The Wald p-value for the test of all genotype interaction terms; i.e. the test of any genotype interaction effect
 Joint.Stat The Wald chi-squared test statistic for the joint test of the genotype term and all of the genotype interaction terms
 Joint.pval The Wald p-value for the joint test of the genotype term and all of the genotype interaction terms; i.e. the test of any genotype effect

When ivars is not NULL, if ivar.return.betaCov is TRUE, then the output is a list with two elements. The first, "results", is the data.frame described above. The second, "betaCov", is a list with length equal to the number of rows of "results", where each element of the list is the covariance matrix of the effect size estimates (betas) for the genotype and genotype interaction terms.

If genoData is a SeqVarData object, the effect size estimate is for each copy of the alternate allele.

Note

The GenotypeData function in the GWASTools package should be used to create the input genoData. Input to the GenotypeData function can easily be created from an R matrix or GDS file. PLINK .bed, .bim, and .fam files can easily be converted to a GDS file with the function snpgdsBED2GDS in the SNPReLate package. Alternatively, the SeqVarData function in the SeqVarTools package can be used to create the input genodata when working with sequencing data.

Author(s)

Matthew P. Conomos

See Also

[fitNullMM](#) for fitting the null mixed model needed as input to `assocTestMM`. [qqPlot](#) for a function to make QQ plots and [manhattanPlot](#) for a function to make Manhattan plots of p-values. [GWASTools](#) for a description of the package containing the following functions: [GenotypeData](#) for a description of creating a `GenotypeData` class object for storing sample and SNP genotype data, [MatrixGenotypeReader](#) for a description of reading in genotype data stored as a matrix, and [GdsGenotypeReader](#) for a description of reading in genotype data stored as a GDS file. Also see [snpgdsBED2GDS](#) in the [SNPRelate](#) package for a description of converting binary PLINK files to GDS.

Examples

```
library(GWASTools)

# file path to GDS file
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
# read in GDS data
HapMap_geno <- GdsGenotypeReader(filename = gdsfile)
# create a GenotypeData class object
HapMap_genoData <- GenotypeData(HapMap_geno)
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")

# run PC-AiR
mypcair <- pcair(genoData = HapMap_genoData, kinMat = HapMap_ASW_MXL_KINGmat,
                divMat = HapMap_ASW_MXL_KINGmat)

# run PC-Relate
mypcrel <- pcrelate(genoData = HapMap_genoData, pcMat = mypcair$vector[,1],
                   training.set = mypcair$unrels)

# generate a phenotype
set.seed(4)
pheno <- 0.2*mypcair$vector[,1] + rnorm(mypcair$nsamp, mean = 0, sd = 1)

# make ScanAnnotationDataFrame
scanAnnot <- ScanAnnotationDataFrame(data.frame(scanID = mypcrel$sample.id,
                                              pc1 = mypcair$vector[,1], pheno = pheno))

# make covMatList
covMatList <- list("Kin" = pcrelateMakeGRM(mypcrel))

# fit the null mixed model
nullmod <- fitNullMM(scanData = scanAnnot, outcome = "pheno", covars = "pc1", covMatList = covMatList)

# run the association test
myassoc <- assocTestMM(genoData = HapMap_genoData, nullMMobj = nullmod)
close(HapMap_genoData)

# make a QQ plot
qqPlot(myassoc$Wald.pval)
```


assocTestSeq

*Aggregate Association Testing with Sequencing Data***Description**

assocTestSeq performs aggregate association tests with sequencing data using the null model fit with `fitNullMM` or `fitNullReg`.

Usage

```
assocTestSeq(seqData, nullModObj, aggVarList, AF.sample = NULL,
             AF.range = c(0,1), weight.beta = c(1, 1), weight.user = NULL,
             test = "Burden", burden.test = "Score", rho = 0,
             pval.method = "davies", verbose = TRUE)
```

Arguments

seqData	An object of class SeqVarData from the package SeqVarTools containing the sequencing genotype data for variants and samples to be used for the analysis.
nullModObj	A null model object returned by <code>fitNullMM</code> when using mixed models or <code>fitNullReg</code> when using linear or logistic regression.
aggVarList	A list specifying the variant aggregation units to be tested; each element of the list represents one aggregate test. Each element of the list should be a data.frame that contains at least two columns: <code>variant.id</code> matching the <code>variant.id</code> in <code>seqData</code> for the variants that should be aggregated, and <code>allele.index</code> specifying which allele in <code>seqData</code> is to be tested at that <code>variant.id</code> . Multiple alleles can be included at the same variant location by including multiple rows with the same <code>variant.id</code> and different <code>allele.index</code> values. <code>allele.index=0</code> indicates the reference allele, <code>allele.index=1</code> is the first alternate allele, <code>allele.index=2</code> is the second alternate allele (for multiallelic variants), and so on up to the number of possible alleles per variant.
AF.sample	A vector of <code>sample.id</code> values specifying which samples should be used for allele frequency calculation. When <code>NULL</code> (the default), all samples included in the test are used. Allele frequency calculation will affect variant inclusion based on <code>AF.range</code> and variant weighting based on <code>weight.beta</code> .
AF.range	A numeric vector of length two specifying the lower and upper bounds on the alternate allele frequency for variants to be included in the analysis. Variants with alternate allele frequencies outside of this range are given a weight of 0 (i.e. excluded).
weight.beta	A numeric vector of length two specifying the two parameters of the Beta distribution used to determine variant weights; weights are given by <code>dbeta(MAF, a, b)</code> , where <code>MAF</code> is the minor allele frequency, and <code>a</code> and <code>b</code> are the two parameters specified here. <code>weight.beta = c(1, 25)</code> gives the Wu weights; <code>weight.beta = c(0.5, 0.5)</code> is proportional to the Madsen-Browning weights; and <code>weight.beta = c(1, 1)</code> gives a weight of 1 to all variants. This input is ignored when <code>weight.user</code> is not <code>NULL</code> .
weight.user	A character string specifying the name of a variable in the <code>variantData</code> slot of the <code>seqData</code> object to be used as variant weights. When left <code>NULL</code> (the default), the weights specified by <code>weight.beta</code> will be used.

test	A character string specifying the type of test to be performed. The possibilities are "Burden" (default) or "SKAT". When this is set to "SKAT" and the parameter rho has multiple values, a SKAT-O test is performed.
burden.test	A character string specifying the type of Burden test to perform when test = "Burden". The possibilities are "Score", "Wald", and "Firth". "Score" can be used for any nullModObj. "Wald" can not be used when the nullModObj is from a mixed model with a binary outcome variable. "Firth" can only be used when the nullModObj is from a logistic regression with a binary outcome variable.
rho	A numeric value (or vector of numeric values) in [0,1] specifying the rho parameter for SKAT. When rho = 0, a standard SKAT test is performed. When rho = 1, a score burden test is performed. When rho is a vector of values, SKAT-O is performed using each of those values as the search space for the optimal rho.
pval.method	A character string specifying which method to use to calculate SKAT p-values. "davies" (the default) uses numerical integration; "kuonen" uses a saddlepoint method; and "liu" uses a moment matching approximation. If the davies method generates an error, kuonen is tried, and then liu as a last resort.
verbose	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Value

A list with the following items:

param	A list with model parameters including:
AF.range	The lower and upper bounds on the alternate allele frequency for variants that were included in the analysis.
weight.beta	The two parameters of the Beta distribution used to determine variant weights if used, NULL otherwise.
weight.user	A character string specifying the name of the variable in the variantData slot of the seqData object used as variant weights if used, NULL otherwise.
family	Either "gaussian" for a continuous outcome or "binomial" for a binary outcome.
mixedmodel	Logical indicating whether or not a mixed model was used to fit the null model.
test	Specifies whether Burden, SKAT, or SKAT-O tests were performed.
burden.test	If test = "Burden", specifies if Score, Wald, or Firth tests were performed.
rho	The values of rho used in the SKAT or SKAT-O test.
pval.method	The p-value calculation method used in SKAT or SKAT-O tests.
nsample	A list with the following values:
analysis	The number of samples included in the analysis.
AF	The number of samples used to calculate allele frequencies.
results	A data.frame containing the results from the main analysis. Each row is a separate aggregate test:
n.site	The number of variant sites included in the test.
n.sample.alt	The number of samples with an observed alternate allele at any variant in the aggregate set.

If test is "Burden":

burden.skew	The skewness of the burden value for all samples.
-------------	---

If `burden.test` is "Score":

<code>Score</code>	The value of the score function
<code>Var</code>	The variance of the score function
<code>Score.stat</code>	The score chi-squared test statistic
<code>Score.pval</code>	The score p-value

If `burden.test` is "Wald":

<code>Est</code>	The effect size estimate for a one unit increase in the burden value
<code>SE</code>	The estimated standard error of the effect size estimate
<code>Wald.stat</code>	The Wald chi-squared test statistic
<code>Wald.pval</code>	The Wald p-value

If `burden.test` is "Firth":

<code>Est</code>	The effect size estimate for a one unit increase in the burden value
<code>SE</code>	The estimated standard error of the effect size estimate
<code>Firth.stat</code>	The Firth test statistic
<code>Firth.pval</code>	The Firth p-value

If `test` is "SKAT":

<code>Q_rho</code>	The SKAT test statistic for the value of <code>rho</code> specified. There will be as many of these variables as there are <code>rho</code> values chosen.
<code>pval_rho</code>	The SKAT p-value for the value of <code>rho</code> specified. There will be as many of these variables as there are <code>rho</code> values chosen.
<code>err_rho</code>	Takes value 1 if there was an error in calculating the p-value for the value of <code>rho</code> specified when using the "kuonen" or "davies" methods; 0 otherwise. When there is an error, the p-value returned is from the "liu" method. There will be as many of these variables as there are <code>rho</code> values chosen.

When `length(rho) > 1` and SKAT-O is performed:

<code>min.pval</code>	The minimum p-value among the p-values calculated for each choice of <code>rho</code> .
<code>opt_rho</code>	The optimal <code>rho</code> value; i.e. the <code>rho</code> value that gave the minimum p-value.
<code>pval_SKATO</code>	The SKAT-O p-value after adjustment for searching across multiple <code>rho</code> values.
<code>variantInfo</code>	A list with as many elements as aggregate tests performed. Each element of the list is a <code>data.frame</code> providing information on the variants used in the aggregate test with results presented in the corresponding row of <code>results</code> . Each of these <code>data.frames</code> has the following information:
<code>variantID</code>	The <code>variant.id</code> value from <code>seqData</code> .
<code>allele</code>	The index of the allele in <code>seqData</code> .
<code>chr</code>	The chromosome the variant is located on.
<code>pos</code>	The position of the variant on the chromosome.
<code>n.obs</code>	The number of samples with observed genotype values at the variant.
<code>freq</code>	The allele frequency calculated using the samples specified by <code>AF.sample</code> (or all samples if <code>AF.sample</code> is <code>NULL</code>) of the alternate allele given by the allele index at the variant.
<code>weight</code>	The weight assigned to the variant in the analysis. A weight of 0 means the variant was excluded.

Author(s)

Matthew P. Conomos

Examples

```

library(SeqVarTools)
library(Biobase)

# open a sequencing GDS file
gdsfile <- seqExampleFileName("gds")
gds <- seqOpen(gdsfile)

# simulate some phenotype data
data(pedigree)
pedigree <- pedigree[match(seqGetData(gds, "sample.id"), pedigree$sample.id),]
pedigree$outcome <- rnorm(nrow(pedigree))

# construct a SeqVarData object
seqData <- SeqVarData(gds, sampleData=AnnotatedDataFrame(pedigree))

# fit the null model
nullmod <- fitNullReg(sampleData(seqData), outcome="outcome", covars="sex")

# select variant aggregation units (allele.index=1 tests the alternate allele)
agg <- list(data.frame(variant.id=1:100, allele.index=1),
           data.frame(variant.id=101:200, allele.index=1),
           data.frame(variant.id=201:300, allele.index=1))

# burden test
assoc <- assocTestSeq(seqData, nullmod, agg, test="Burden")
assoc$results
lapply(assoc$variantInfo, head)

# SKAT test
assoc <- assocTestSeq(seqData, nullmod, agg, test="SKAT")
assoc$results

# SKAT-0 test
assoc <- assocTestSeq(seqData, nullmod, agg, test="SKAT", rho=seq(0, 1, 0.25))
assoc$results

# user-specified weights
variant.id <- seqGetData(gds, "variant.id")
weights <- data.frame(variant.id, weight=runif(length(variant.id)))
variantData(seqData) <- AnnotatedDataFrame(weights)
assoc <- assocTestSeq(seqData, nullmod, agg, test="Burden", weight.user="weight")
assoc$results
lapply(assoc$variantInfo, head)

seqClose(seqData)

```

Description

assocTestSeqWindow performs aggregate association tests with sequencing data in sliding windows using the null model fit with `fitNullMM` or `fitNullReg`.

Usage

```
assocTestSeqWindow(seqData, nullModObj, variant.include = NULL, chromosome = NULL,
  window.size = 50, window.shift = 20, AF.sample = NULL, AF.range = c(0,1),
  weight.beta = c(1, 1), weight.user = NULL, test = "Burden",
  burden.test = "Score", rho = 0, pval.method = "davies", verbose = TRUE)
```

Arguments

<code>seqData</code>	An object of class <code>SeqVarData</code> from the package <code>SeqVarTools</code> containing the sequencing genotype data for variants and samples to be used for the analysis.
<code>nullModObj</code>	A null model object returned by <code>fitNullMM</code> when using mixed models or <code>fitNullReg</code> when using linear or logistic regression.
<code>variant.include</code>	A vector of <code>variant.id</code> values indicating which variants to include in the sliding window analysis. If <code>NULL</code> , see <code>chromosome</code> for further details.
<code>chromosome</code>	A vector specifying which chromosomes to analyze. This parameter is only considered when <code>variant.include</code> is <code>NULL</code> ; if <code>chromosome</code> is also <code>NULL</code> , then all variants in <code>seqData</code> are included.
<code>window.size</code>	The size, in kb, of each window to be analyzed. Windows are defined based on physical position.
<code>window.shift</code>	The distance, in kb, that the window is shifted to create each new window.
<code>AF.sample</code>	A vector of <code>sample.id</code> values specifying which samples should be used for allele frequency calculation. When <code>NULL</code> (the default), all samples included in the test are used. Allele frequency calculation will affect variant inclusion based on <code>AF.range</code> and variant weighting based on <code>weight.beta</code> .
<code>AF.range</code>	A numeric vector of length two specifying the lower and upper bounds on the alternate allele frequency for variants to be included in the analysis. Variants with alternate allele frequencies outside of this range are given a weight of 0 (i.e. excluded).
<code>weight.beta</code>	A numeric vector of length two specifying the two parameters of the Beta distribution used to determine variant weights; weights are given by <code>dbeta(MAF, a, b)</code> , where <code>MAF</code> is the minor allele frequency, and <code>a</code> and <code>b</code> are the two parameters specified here. <code>weight.beta = c(1, 25)</code> gives the Wu weights; <code>weight.beta = c(0.5, 0.5)</code> is proportional to the Madsen-Browning weights; and <code>weight.beta = c(1, 1)</code> gives a weight of 1 to all variants. This input is ignored when <code>weight.user</code> is not <code>NULL</code> .
<code>weight.user</code>	A character string specifying the name of a variable in the <code>variantData</code> slot of the <code>seqData</code> object to be used as variant weights. When left <code>NULL</code> (the default), the weights specified by <code>weight.beta</code> will be used.
<code>test</code>	A character string specifying the type of test to be performed. The possibilities are "Burden" (default) or "SKAT". When this is set to "SKAT" and the parameter <code>rho</code> has multiple values, a SKAT-O test is performed.
<code>burden.test</code>	A character string specifying the type of Burden test to perform when <code>test = "Burden"</code> . The possibilities are "Score", "Wald", and "Firth". "Score" can be

used for any nullModObj. "Wald" can not be used when the nullModObj is from a mixed model with a binary outcome variable. "Firth" can only be used when the nullModObj is from a logistic regression with a binary outcome variable.

rho	A numeric value (or vector of numeric values) in [0,1] specifying the rho parameter for SKAT. When rho = 0, a standard SKAT test is performed. When rho = 1, a score burden test is performed. When rho is a vector of values, SKAT-O is performed using each of those values as the search space for the optimal rho.
pval.method	A character string specifying which method to use to calculate SKAT p-values. "davies" (the default) uses numerical integration; "kuonen" uses a saddlepoint method; and "liu" uses a moment matching approximation. If the davies method generates an error, kuonen is tried, and then liu as a last resort.
verbose	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Value

A list with the following items:

param	A list with model parameters including:
AF.range	The lower and upper bounds on the alternate allele frequency for variants that were included in the analysis.
weight.beta	The two parameters of the Beta distribution used to determine variant weights if used, NULL otherwise.
weight.user	A character string specifying the name of the variable in the variantData slot of the seqData object used as variant weights if used, NULL otherwise.
family	Either "gaussian" for a continuous outcome or "binomial" for a binary outcome.
mixedmodel	Logical indicating whether or not a mixed model was used to fit the null model.
test	Specifies whether Burden, SKAT, or SKAT-O tests were performed.
burden.test	If test = "Burden", specifies if Score, Wald, or Firth tests were performed.
rho	The values of rho used in the SKAT or SKAT-O test.
pval.method	The p-value calculation method used in SKAT or SKAT-O tests.
window	A list with the following values:
size	The size of the windows in kb
shift	The distance each window was shifted, in kb, to create the next window.
nsample	A list with the following values:
analysis	The number of samples included in the analysis.
AF	The number of samples used to calculate allele frequencies.
results	A data.frame containing the results from the main analysis. Each row is a separate aggregate test:
chr	The chromosome that the window is on.
window.start	The base position of the start of the window.
window.stop	The base position of the end of the window.
n.site	The number of variant sites included in the test.
dup	Takes the value 1 if the variants in this window are identical to the variants in the previous window; takes the value 0 otherwise.

If test is "Burden":

burden.skew The skewness of the burden value for all samples.

If burden.test is "Score":

Score The value of the score function
 Var The variance of the score function
 Score.stat The score chi-squared test statistic
 Score.pval The score p-value

If burden.test is "Wald":

Est The effect size estimate for a one unit increase in the burden value
 SE The estimated standard error of the effect size estimate
 Wald.stat The Wald chi-squared test statistic
 Wald.pval The Wald p-value

If burden.test is "Firth":

Est The effect size estimate for a one unit increase in the burden value
 SE The estimated standard error of the effect size estimate
 Firth.stat The Firth test statistic
 Firth.pval The Firth p-value

If test is "SKAT":

Q_rho The SKAT test statistic for the value of rho specified. There will be as many of these variables as there are rho values chosen.
 pval_rho The SKAT p-value for the value of rho specified. There will be as many of these variables as there are rho values chosen.
 err_rho Takes value 1 if there was an error in calculating the p-value for the value of rho specified when using the "kuonen" or "davies" methods; 0 otherwise. When there is an error, the p-value returned is from the "liu" method. There will be as many of these variables as there are rho values chosen.

When length(rho) > 1 and SKAT-O is performed:

min.pval The minimum p-value among the p-values calculated for each choice of rho.
 opt.rho The optimal rho value; i.e. the rho value that gave the minimum p-value.
 pval_SKATO The SKAT-O p-value after adjustment for searching across multiple rho values.
 variantInfo A data.frame providing information on all of the variants used in the aggregate tests across all of the windows. The data.frame contains the following information:
 variantID The variant.id value from seqData.
 allele The index of the allele in seqData.
 chr The chromosome the variant is located on.
 pos The position of the variant on the chromosome.
 n.obs The number of samples with observed genotype values at the variant.
 freq The allele frequency calculated using the samples specified by AF.sample (or all samples if AF.sample is NULL) of the alternate allele given by the allele index at the variant.
 weight The weight assigned to the variant in the analysis. A weight of 0 means the variant was excluded.

Author(s)

Matthew P. Conomos

Examples

```

library(SeqVarTools)
library(Biobase)

# open a sequencing GDS file
gdsfile <- seqExampleFileName("gds")
gds <- seqOpen(gdsfile)

# simulate some phenotype data
data(pedigree)
pedigree <- pedigree[match(seqGetData(gds, "sample.id"), pedigree$sample.id),]
pedigree$outcome <- rnorm(nrow(pedigree))

# construct a SeqVarData object
seqData <- SeqVarData(gds, sampleData=AnnotatedDataFrame(pedigree))

# fit the null model
nullmod <- fitNullReg(sampleData(seqData), outcome="outcome", covars="sex")

# burden test
assoc <- assocTestSeqWindow(seqData, nullmod, chromosome=22, test="Burden")
head(assoc$results)
head(assoc$variantInfo)

# SKAT test
assoc <- assocTestSeqWindow(seqData, nullmod, chromosome=22, test="SKAT")
head(assoc$results)

# SKAT-O test
assoc <- assocTestSeqWindow(seqData, nullmod, chromosome=22, test="SKAT", rho=seq(0, 1, 0.25))
head(assoc$results)

# user-specified weights
variant.id <- seqGetData(gds, "variant.id")
weights <- data.frame(variant.id, weight=runif(length(variant.id)))
variantData(seqData) <- AnnotatedDataFrame(weights)
assoc <- assocTestSeqWindow(seqData, nullmod, chromosome=22, test="Burden", weight.user="weight")
head(assoc$results)
head(assoc$variantInfo)

seqClose(seqData)

```

Description

fitNullMM fits a mixed model with random effects specified by their covariance structures; this allows for the inclusion of a polygenic random effect using a kinship matrix or genetic relationship matrix (GRM). The output of fitNullMM can be used to estimate genetic heritability and can be passed to [assocTestMM](#) for the purpose of genetic association testing.

Usage

```
fitNullMM(scanData, outcome, covars = NULL, covMatList, scan.include = NULL,
          family = gaussian, group.var = NULL, start = NULL,
          AIREML.tol = 1e-6, maxIter = 100, dropZeros = TRUE, verbose = TRUE)
```

Arguments

scanData	An object of class ScanAnnotationDataFrame from the package GWASTools, AnnotatedDataFrame, or class data.frame containing the outcome and covariate data for the samples to be used for the analysis. scanData must have a column scanID containing unique IDs for all samples.
outcome	A character string specifying the name of the outcome variable in scanData.
covars	A vector of character strings specifying the names of the fixed effect covariates in scanData; an intercept term is automatically included. If NULL (default) the only fixed effect covariate is the intercept term.
covMatList	A list of matrices specifying the covariance structures of the random effects terms. The column and row names of each of these matrices must match the scanIDs from scanData. If only one random effect is being used, a single matrix (not in a list) can be used. See 'Details' for more information.
scan.include	A vector of scanIDs for samples to include in the analysis. If NULL, all samples in scanData are included.
family	A description of the error distribution to be used in the model. The default "gaussian" fits a linear mixed model; see family for further options, and see 'Details' for more information.
group.var	This variable can only be used when family = gaussian. A character string specifying the name of a categorical variable in scanData that is used to fit heterogeneous residual error variances. If NULL (default), then a standard LMM with constant residual variance for all samples is fit. See 'Details' for more information.
start	A vector of starting values for the variance component estimation procedure. The function will pick reasonable starting values when left NULL (default). See 'Details' for more information.
AIREML.tol	The convergence threshold for the Average Information REML (AIREML) procedure used to estimate the variance components of the random effects. See 'Details' for more information.
maxIter	The maximum number of iterations allowed in the AIREML procedure.
dropZeros	Logical indicator of whether variance component terms that converge to 0 should be removed from the model; the default is TRUE. See 'Details' for more information.
verbose	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Details

covMatList is used to specify the covariance structures of the random effects terms in the model. For example, to include a polygenic random effect, one matrix in covMatList could be a kinship matrix or a genetic relationship matrix (GRM). As another example, to include household membership as a random effect, one matrix in covMatList should be a 0/1 matrix with a 1 in the [i,j] and

[j,i] entries if individuals *i* and *j* are in the same household and 0 otherwise; the diagonals of such a matrix should all be 1.

When `family` is not gaussian, the penalized quasi-likelihood (PQL) approximation to the generalized linear mixed model (GLMM) is fit following the procedure of GMMAT (Chen et al.).

For some outcomes, there may be evidence that different groups of observations have different residual variances, and the standard LMM assumption of homoscedasticity is violated. When `group.var` is specified, separate (heterogeneous) residual variance components are fit for each unique value of `group.var`.

Let *m* be the number of matrices in `covMatList` and let *g* be the number of categories in the variable specified by `group.var`. The length of the `start` vector must be (*m* + 1) when `family` is gaussian and `group.var` is NULL; (*m* + *g*) when `family` is gaussian and `group.var` is specified; or *m* when `family` is not gaussian.

A Newton-Raphson iterative procedure with Average Information REML (AIREML) is used to estimate the variance components of the random effects. When the Euclidean distance between the new and previous variance component estimates is less than `AIREML.tol`, the algorithm declares convergence of the estimates. Sometimes a variance component may approach the boundary of the parameter space at 0; step-halving is used to prevent any component from becoming negative. However, when a variance component gets near the 0 boundary, the algorithm can sometimes get "stuck", preventing the other variance components from converging; if `dropZeros` is TRUE, then variance components that converge to a value less than `AIREML.tol` will be dropped from the model and the estimation procedure will continue with the remaining variance components.

Value

An object of class 'GENESIS.nullMixedModel'. A list including:

<code>varComp</code>	The variance component estimates. There is one variance component for each random effect specified in <code>covMatList</code> . When <code>family</code> is gaussian, there are additional residual variance components; one residual variance component when <code>group.var</code> is NULL, and as many residual variance components as there are unique values of <code>group.var</code> when it is specified.
<code>varCompCov</code>	The estimated covariance matrix of the variance component estimates given by <code>varComp</code> . This can be used for hypothesis tests regarding the variance components.
<code>fixef</code>	A data.frame with effect size estimates (betas), standard errors, chi-squared test statistics, and p-values for each of the fixed effect covariates specified in <code>covars</code> .
<code>betaCov</code>	The estimated covariance matrix of the effect size estimates (betas) of the fixed effect covariates. This can be used for hypothesis tests regarding the fixed effects.
<code>fitted.values</code>	The fitted values from the mixed model; i.e. $W \cdot \text{beta}$ where <i>W</i> is the design matrix and <i>beta</i> are the effect size estimates for the fixed effects.
<code>resid.marginal</code>	The marginal residuals from the mixed model; i.e. $Y - W \cdot \text{beta}$ where <i>Y</i> is the vector of outcome values.
<code>eta</code>	The linear predictor from the mixed model; i.e. $W \cdot \text{beta} + Z \cdot u$ where $Z \cdot u$ specifies the effects of the random effects.
<code>resid.conditional</code>	The conditional residuals from the mixed model; i.e. $Y - W \cdot \text{beta} - Z \cdot u$.
<code>logLikR</code>	The restricted log-likelihood value.
<code>logLik</code>	The log-likelihood value.

AIC	The Akaike Information Criterion value.
RSS	The residual sum of squares from the model fit. When family is gaussian, this will typically be 1 since the residual variance component is estimated separately.
workingY	The "working" outcome vector. When family is gaussian, this is just the original outcome vector. When family is not gaussian, this is the PQL linearization of the outcome vector. This is used by <code>assocTestMM</code> for genetic association testing. See 'Details' for more information.
outcome	The original outcome vector. When family is gaussian, this is equal to <code>workingY</code> .
<code>model.matrix</code>	The design matrix for the fixed effect covariates used in the model.
<code>cholSigmaInv</code>	The Cholesky decomposition of the inverse of the estimated outcome covariance structure. This is used by <code>assocTestMM</code> for genetic association testing.
<code>scanID</code>	A vector of scanIDs for the samples used in the analysis.
<code>family</code>	A character string specifying the family used in the analysis.
<code>converged</code>	A logical indicator of whether the AIREML procedure for estimating the random effects variance components converged.
<code>zeroFLAG</code>	A vector of logicals the same length as <code>varComp</code> specifying whether the corresponding variance component estimate was set to 0 by the function due to convergence to the boundary in the AIREML procedure.
<code>hetResid</code>	A logical indicator of whether heterogeneous residual variance components were used in the model (specified by <code>group.var</code>).
<code>call</code>	The call to <code>fitNullMM</code> .

Author(s)

Matthew P. Conomos

References

- Chen H, Wang C, Conomos MP, Stilp AM, Li Z, Sofer T, Szpiro AA, Chen W, Brehm JM, Celedon JC, Redline S, Papanicolaou GJ, Thornton TA, Laurie CC, Rice K and Lin X. Control for Population Structure and Relatedness for Binary Traits in Genetic Association Studies Using Logistic Mixed Models. (Submitted).
- Breslow NE and Clayton DG. (1993). Approximate Inference in Generalized Linear Mixed Models. *Journal of the American Statistical Association* 88: 9-25.
- Gilmour, A.R., Thompson, R., & Cullis, B.R. (1995). Average information REML: an efficient algorithm for variance parameter estimation in linear mixed models. *Biometrics*, 1440-1450.
- Gogarten, S.M., Bhangale, T., Conomos, M.P., Laurie, C.A., McHugh, C.P., Painter, I., ... & Laurie, C.C. (2012). GWASTools: an R/Bioconductor package for quality control and analysis of Genome-Wide Association Studies. *Bioinformatics*, 28(24), 3329-3331.

See Also

[varCompCI](#) for estimating confidence intervals for the variance components and the proportion of variability (heritability) they explain, [assocTestMM](#) for running mixed model genetic association tests using the output from `fitNullMM`. [GWASTools](#) for a description of the package containing the `ScanAnnotationDataFrame` class.

Examples

```

library(GWASTools)

# file path to GDS file
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
# read in GDS data
HapMap_geno <- GdsGenotypeReader(filename = gdsfile)
# create a GenotypeData class object
HapMap_genoData <- GenotypeData(HapMap_geno)
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")

# run PC-AiR
mypcair <- pcair(genoData = HapMap_genoData, kinMat = HapMap_ASW_MXL_KINGmat,
                divMat = HapMap_ASW_MXL_KINGmat)

# run PC-Relate
mypcrel <- pcrelate(genoData = HapMap_genoData, pcMat = mypcair$vectors[,1],
                  training.set = mypcair$unrels)
close(HapMap_genoData)

# generate a phenotype
set.seed(4)
pheno <- 0.2*mypcair$vectors[,1] + rnorm(mypcair$nsamp, mean = 0, sd = 1)

# make ScanAnnotationDataFrame
scanAnnot <- ScanAnnotationDataFrame(data.frame(scanID = mypcrel$sample.id,
        pc1 = mypcair$vectors[,1], pheno = pheno))

# make covMatList
covMatList <- list("Kin" = pcrelateMakeGRM(mypcrel))

# fit the null mixed model
nullmod <- fitNullMM(scanData = scanAnnot, outcome = "pheno", covars = "pc1", covMatList = covMatList)

```

fitNullReg

*Fit a Regression Model Under the Null Hypothesis***Description**

fitNullReg fits a regression model. The output of fitNullReg can be passed to [assocTestSeq](#) or [assocTestSeqWindow](#) for the purpose of genetic association testing.

Usage

```
fitNullReg(scanData, outcome, covars = NULL, scan.include = NULL,
          family = gaussian, verbose = TRUE)
```

Arguments

scanData An object of class ScanAnnotationDataFrame from the package GWASTools, AnnotatedDataFrame, or class data.frame containing the outcome and covariate data for the samples to be used for the analysis. scanData must have a column scanID containing unique IDs for all samples.

outcome	A character string specifying the name of the outcome variable in scanData.
covars	A vector of character strings specifying the names of the fixed effect covariates in scanData; an intercept term is automatically included. If NULL (default) the only fixed effect covariate is the intercept term.
scan.include	A vector of scanIDs for samples to include in the analysis. If NULL, all samples in scanData are included.
family	A description of the error distribution to be used in the model. The default "gaussian" fits a linear model; see family for further options.
verbose	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Value

An object of class 'GENESIS.nullModel'. A list including:

fixef	A data.frame with effect size estimates (betas), standard errors, chi-squared test statistics, and p-values for each of the fixed effect covariates specified in covars.
betaCov	The estimated covariance matrix of the effect size estimates (betas) of the fixed effect covariates. This can be used for hypothesis tests regarding the fixed effects.
resid.response	The residuals from the model.
logLik	The log-likelihood value.
AIC	The Akaike Information Criterion value.
workingY	The "working" outcome vector. When family is gaussian, this is just the original outcome vector. When family is not gaussian, this is the PQL linearization of the outcome vector. This is used by <code>assocTestSeq</code> for genetic association testing.
model.matrix	The design matrix for the fixed effect covariates used in the model.
aliased	Coefficients removed from the model.
sigma	Variance of the model.
scanID	A vector of scanIDs for the samples used in the analysis.
family	A character string specifying the family used in the analysis.

Author(s)

Matthew P. Conomos

HapMap_ASW_MXL_KINGmat

Matrix of Pairwise Kinship Coefficient Estimates for the combined HapMap ASW and MXL Sample found with the KING-robust estimator from the KING software.

Description

KING-robust kinship coefficient estimates for the combined HapMap African Americans in the Southwest U.S. (ASW) and Mexican Americans in Los Angeles (MXL) samples.

Usage

```
data(HapMap_ASW_MXL_KINGmat)
```

Format

The format is: num [1:173, 1:173] 0 0.00157 -0.00417 0.00209 0.00172 ...

Value

A matrix of pairwise kinship coefficient estimates as calculated with KING-robust for the combined HapMap African Americans in the Southwest U.S. (ASW) and Mexican Americans in Los Angeles (MXL) samples.

Source

<http://hapmap.ncbi.nlm.nih.gov/>

References

International HapMap 3 Consortium. (2010). Integrating common and rare genetic variation in diverse human populations. *Nature*, 467(7311), 52-58.

king2mat

Convert KING text output to an R Matrix

Description

king2mat is used to extract the pairwise kinship coefficient estimates or IBS0 values from the output text files of KING and put them into an R object of class `matrix` that can be read by the functions `pcair` and `pcairPartition`.

Usage

```
king2mat(file.kin0, file.kin = NULL, iids = NULL,
         type = "kinship", verbose = TRUE)
```

Arguments

<code>file.kin0</code>	File name of the <code>.kin0</code> text file output from KING.
<code>file.kin</code>	Optional file name of the <code>.kin</code> text file output from KING.
<code>iids</code>	An optional vector of individual IDs in the same order as desired for the output matrix. See 'Details' for more information.
<code>type</code>	Character string taking the values "kinship" (default) or "IBS0", to inform the function to read in kinship coefficients or IBS0 values from the KING output.
<code>verbose</code>	A logical indicating whether or not to print status updates to the console; the default is TRUE.

Details

When using the function `pcair`, it is important that the order of individuals in the `kinMat` matrix matches the order of individuals in `genoData`. The KING software has a tendency to reorder individuals. If `iids = NULL`, the default is for the order to be taken from the KING output text file. By specifying `iids` the user can control the order of individuals in the output matrix. The IDs used for `iids` must be the same set of character IDs that are output as columns 'ID1' and 'ID2' in the KING output text files; all of the IDs specified in `iids` must be in the KING output, and all IDs in the KING output must be specified in `iids`.

Value

An object of class 'matrix' with pairwise kinship coefficients or IBS0 values as estimated by KING for each pair of individuals in the sample. The estimates are on both the upper and lower triangle of the matrix, and the diagonal is arbitrarily set to 0.5. Individual IDs are set as the column and row names of the matrix.

Author(s)

Matthew P. Conomos

References

Conomos M.P., Miller M., & Thornton T. (2015). Robust Inference of Population Structure for Ancestry Prediction and Correction of Stratification in the Presence of Relatedness. *Genetic Epidemiology*, 39(4), 276-293.

Manichaikul, A., Mychaleckyj, J.C., Rich, S.S., Daly, K., Sale, M., & Chen, W.M. (2010). Robust relationship inference in genome-wide association studies. *Bioinformatics*, 26(22), 2867-2873.

See Also

[pcair](#) and [pcairPartition](#) for functions that use the output matrix.

Examples

```
file.kin0 <- system.file("extdata", "MXL_ASW.kin0", package="GENESIS")
file.kin <- system.file("extdata", "MXL_ASW.kin", package="GENESIS")
KINGmat <- king2mat(file.kin0 = file.kin0, file.kin = file.kin, type="kinship")
```

Description

`pcair` is used to perform a Principal Components Analysis using genome-wide SNP data for the detection of population structure in a sample. Unlike a standard PCA, PC-AiR accounts for sample relatedness (known or cryptic) to provide accurate ancestry inference that is not confounded by family structure.

Usage

```
pcair(genoData, v = 20, kinMat = NULL, kin.thresh = 2^(-11/2),
      divMat = NULL, div.thresh = -2^(-11/2), unrel.set = NULL,
      scan.include = NULL, snp.include = NULL, chromosome = NULL,
      snp.block.size = 10000, MAF = 0.01, verbose = TRUE)
## S3 method for class 'pcair'
print(x, ...)
## S3 method for class 'pcair'
summary(object, ...)
## S3 method for class 'summary.pcair'
print(x, ...)
```

Arguments

genoData	An object of class <code>GenotypeData</code> from the package <code>GWASTools</code> containing the genotype data for SNPs and samples to be used for the analysis. This object can easily be created from a matrix of SNP genotype data, PLINK files, or GDS files. Alternatively, this could be an object of class <code>SeqVarData</code> from the package <code>SeqVarTools</code> containing the genotype data for the sequencing variants and samples to be used for the analysis.
v	The number of principal components to be returned; the default is 20. If <code>v = NULL</code> , then all the principal components are returned.
kinMat	An optional symmetric matrix of pairwise kinship coefficients for every pair of individuals in the sample (the values on the diagonal do not matter, but the upper and lower triangles must both be filled) used for partitioning the sample into the 'unrelated' and 'related' subsets. See 'Details' for how this interacts with <code>kin.thresh</code> and <code>unrel.set</code> . IDs for each individual must be set as the row and column names of the matrix.
kin.thresh	Threshold value on <code>kinMat</code> used for declaring each pair of individuals as related or unrelated. The default value is $2^{(-11/2)} \sim 0.022$. See 'Details' for how this interacts with <code>kinMat</code> .
divMat	An optional symmetric matrix of pairwise divergence measures for every pair of individuals in the sample (the values on the diagonal do not matter, but the upper and lower triangles must both be filled) used for partitioning the sample into the 'unrelated' and 'related' subsets. See 'Details' for how this interacts with <code>div.thresh</code> . IDs for each individual must be set as the row and column names of the matrix.
div.thresh	Threshold value on <code>divMat</code> used for deciding if each pair of individuals is ancestrally divergent. The default value is $-2^{(-11/2)} \sim -0.022$. See 'Details' for how this interacts with <code>divMat</code> .
unrel.set	An optional vector of IDs for identifying individuals that are forced into the unrelated subset. See 'Details' for how this interacts with <code>kinMat</code> .
scan.include	A vector of IDs for samples to include in the analysis. If <code>NULL</code> , all samples are included.
snp.include	A vector of SNP IDs to include in the analysis. If <code>NULL</code> , see <code>chromosome</code> for further details.
chromosome	A vector of integers specifying which chromosomes to analyze. This parameter is only considered when <code>snp.include</code> is <code>NULL</code> ; if <code>chromosome</code> is also <code>NULL</code> , then all SNPs are included.

snp.block.size	The number of SNPs to read-in/analyze at once. The default value is 10000.
MAF	Minor allele frequency filter; any SNPs with MAF less than this value will be excluded from the analysis; the default value is 0.01.
verbose	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.
object	An object of class 'pcair', i.e. output from the pcair function.
x	An object of class 'pcair', i.e. output from the pcair function.
...	Further arguments passed to or from other methods.

Details

The basic premise of PC-AiR is to partition the entire sample of individuals into an ancestry representative 'unrelated subset' and a 'related set', perform standard PCA on the 'unrelated subset', and predict PC values for the 'related subset'.

We recommend using software that accounts for population structure to estimate pairwise kinship coefficients to be used in kinMat. Any pair of individuals with a pairwise kinship greater than kin.thresh will be declared 'related.' Kinship coefficient estimates from the KING-robust software are used as measures of ancestry divergence in divMat. Any pair of individuals with a pairwise divergence measure less than div.thresh will be declared ancestrally 'divergent'. Typically, kin.thresh and div.thresh are set to be the amount of error around 0 expected in the estimate for a pair of truly unrelated individuals.

If divMat = NULL and kinMat is specified, the kinship coefficient estimates in kinMat will also be used as divergence measures in place of divMat.

It is important that the order of individuals in the matrices kinMat and divMat match the order of individuals in the genoData.

There are multiple ways to partition the sample into an ancestry representative 'unrelated subset' and a 'related subset'. If kinMat is specified and unrel.set = NULL, then the PC-AiR algorithm is used to find an 'optimal' partition (see 'References' for a paper describing the algorithm). If kinMat = NULL and unrel.set is specified, then the individuals with IDs in unrel.set are used as the 'unrelated subset'. If both kinMat and unrel.set are specified, then all individuals with IDs in unrel.set are forced in the 'unrelated subset' and the PC-AiR algorithm is used to partition the rest of the sample; this is especially useful for including reference samples of known ancestry in the 'unrelated subset'. If kinMat = NULL and unrel.set = NULL, then a standard principal components analysis that does not account for relatedness is performed.

Value

An object of class 'pcair'. A list including:

vectors	A matrix of the top v principal components; each column is a principal component. Sample IDs are provided as rownames.
values	A vector of eigenvalues matching the top v principal components. These values are determined from the standard PCA run on the 'unrelated subset'.
sum.values	The sum of all the eigenvalues from the standard PCA run on the 'unrelated subset' (regardless of how many were returned).
rels	A vector of IDs for individuals in the 'related subset'.
unrels	A vector of IDs for individuals in the 'unrelated subset'.
kin.thresh	The threshold value used for declaring each pair of individuals as related or unrelated.

div.thresh	The threshold value used for determining if each pair of individuals is ancestrally divergent.
nsamp	The total number of samples in the analysis.
nsnps	The total number of SNPs used in the analysis, after filtering on MAF.
MAF	The minor allele frequency (MAF) filter used on SNPs.
call	The function call passed to <code>pcair</code> .
method	A character string. Either "PC-AiR" or "Standard PCA" identifying which method was used for computing principal components.

Note

The `GenotypeData` function in the `GWASTools` package should be used to create the input `genoData`. Input to the `GenotypeData` function can easily be created from an R matrix or GDS file. PLINK `.bed`, `.bim`, and `.fam` files can easily be converted to a GDS file with the function `snpGdsBED2GDS` in the `SNPRelate` package. Alternatively, the `SeqVarData` function in the `SeqVarTools` package can be used to create the input `genodata` when working with sequencing data.

Author(s)

Matthew P. Conomos

References

- Conomos M.P., Miller M., & Thornton T. (2015). Robust Inference of Population Structure for Ancestry Prediction and Correction of Stratification in the Presence of Relatedness. *Genetic Epidemiology*, 39(4), 276-293.
- Gogarten, S.M., Bhangale, T., Conomos, M.P., Laurie, C.A., McHugh, C.P., Painter, I., ... & Laurie, C.C. (2012). GWASTools: an R/Bioconductor package for quality control and analysis of Genome-Wide Association Studies. *Bioinformatics*, 28(24), 3329-3331.
- Manichaikul, A., Mychaleckyj, J.C., Rich, S.S., Daly, K., Sale, M., & Chen, W.M. (2010). Robust relationship inference in genome-wide association studies. *Bioinformatics*, 26(22), 2867-2873.

See Also

[pcairPartition](#) for a description of the function used by `pcair` that can be used to partition the sample into 'unrelated' and 'related' subsets without performing PCA. [plot.pcair](#) for plotting. [king2mat](#) for creating a matrix of pairwise kinship coefficient estimates from KING output text files that can be used for `kinMat` or `divMat`. [GWASTools](#) for a description of the package containing the following functions: [GenotypeData](#) for a description of creating a `GenotypeData` class object for storing sample and SNP genotype data, [MatrixGenotypeReader](#) for a description of reading in genotype data stored as a matrix, and [GdsGenotypeReader](#) for a description of reading in genotype data stored as a GDS file. Also see [snpGdsBED2GDS](#) in the `SNPRelate` package for a description of converting binary PLINK files to GDS. The generic functions [summary](#) and [print](#).

Examples

```
library(GWASTools)

# file path to GDS file
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
# read in GDS data
HapMap_geno <- GdsGenotypeReader(filename = gdsfile)
```

```

# create a GenotypeData class object
HapMap_genodata <- GenotypeData(HapMap_geno)
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")
# run PC-AiR
mypcair <- pcair(genodata = HapMap_genodata, kinMat = HapMap_ASW_MXL_KINGmat,
                 divMat = HapMap_ASW_MXL_KINGmat)
close(HapMap_genodata)

```

pcairPartition	<i>Partition a sample into an ancestry representative 'unrelated subset' and a 'related subset'</i>
----------------	---

Description

pcairPartition is used to partition a sample from a genetic study into an ancestry representative 'unrelated subset' and a 'related subset'. The 'unrelated subset' contains individuals who are all mutually unrelated to each other and representative of the ancestries of all individuals in the sample, and the 'related subset' contains individuals who are related to someone in the 'unrelated subset'.

Usage

```

pcairPartition(kinMat, kin.thresh = 2-11/2, divMat = NULL,
               div.thresh = -2-11/2, unrel.set = NULL)

```

Arguments

kinMat	A symmetric matrix of pairwise kinship coefficients for every pair of individuals in the sample (the values on the diagonal do not matter, but the upper and lower triangles must both be filled) used for partitioning the sample into the 'unrelated' and 'related' subsets. See 'Details' for how this interacts with kin.thresh and unrel.set. IDs for each individual must be set as the row and column names of the matrix.
kin.thresh	Threshold value on kinMat used for declaring each pair of individuals as related or unrelated. The default value is 0.025. See 'Details' for how this interacts with kinMat.
divMat	A symmetric matrix of pairwise ancestry divergence measures for every pair of individuals in the sample (the values on the diagonal do not matter, but the upper and lower triangles must both be filled) used for partitioning the sample into the 'unrelated' and 'related' subsets. See 'Details' for how this interacts with div.thresh. IDs for each individual must be set as the row and column names of the matrix.
div.thresh	Threshold value on divMat used for deciding if each pair of individuals is ancestrally divergent. The default value is -0.025. See 'Details' for how this interacts with divMat.
unrel.set	An optional vector of IDs for identifying individuals that are forced into the unrelated subset. See 'Details' for how this interacts with kinMat.

Details

We recommend using software that accounts for population structure to estimate pairwise kinship coefficients to be used in `kinMat`. Any pair of individuals with a pairwise kinship greater than `kin.thresh` will be declared 'related.' Kinship coefficient estimates from the KING-robust software are typically used as measures of ancestry divergence in `divMat`. Any pair of individuals with a pairwise divergence measure less than `div.thresh` will be declared ancestrally 'divergent'. Typically, `kin.thresh` and `div.thresh` are set to be the amount of error around 0 expected in the estimate for a pair of truly unrelated individuals. If `unrel.set = NULL`, the PC-AiR algorithm is used to find an 'optimal' partition (see 'References' for a paper describing the algorithm). If `unrel.set` and `kinMat` are both specified, then all individuals with IDs in `unrel.set` are forced in the 'unrelated subset' and the PC-AiR algorithm is used to partition the rest of the sample; this is especially useful for including reference samples of known ancestry in the 'unrelated subset'.

Value

A list including:

<code>rels</code>	A vector of IDs for individuals in the 'related subset'.
<code>unrels</code>	A vector of IDs for individuals in the 'unrelated subset'.

Note

`pcairPartition` is called internally in the function `pcair` but may also be used on its own to partition the sample into an ancestry representative 'unrelated' subset and a 'related' subset without performing PCA.

Author(s)

Matthew P. Conomos

References

Conomos M.P., Miller M., & Thornton T. (2015). Robust Inference of Population Structure for Ancestry Prediction and Correction of Stratification in the Presence of Relatedness. *Genetic Epidemiology*, 39(4), 276-293.

Manichaikul, A., Mychaleckyj, J.C., Rich, S.S., Daly, K., Sale, M., & Chen, W.M. (2010). Robust relationship inference in genome-wide association studies. *Bioinformatics*, 26(22), 2867-2873.

See Also

`pcair` which uses this function for finding principal components in the presence of related individuals. `king2mat` for creating a matrix of kinship coefficient estimates or pairwise ancestry divergence measures from KING output text files that can be used as `kinMat` or `divMat`.

Examples

```
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")
# partition the sample
part <- pcairPartition(kinMat = HapMap_ASW_MXL_KINGmat,
divMat = HapMap_ASW_MXL_KINGmat)
```

pcrelate

*PC-Relate: Model-Free Estimation of Recent Genetic Relatedness***Description**

pcrelate is used to estimate kinship coefficients, IBD sharing probabilities, and inbreeding coefficients using genome-wide SNP data. PC-Relate accounts for population structure (ancestry) among sample individuals through the use of ancestry representative principal components (PCs) to provide accurate relatedness estimates due only to recent family (pedigree) structure.

Usage

```
pcrelate(genoData, pcMat = NULL, freq.type = "individual", scale = "overall",
         ibd.probs = TRUE, scan.include = NULL, training.set = NULL, scan.block.size = 5000,
         snp.include = NULL, chromosome = NULL, snp.block.size = 10000,
         MAF = 0.01, write.to.gds = FALSE, gds.prefix = NULL,
         correct = TRUE, verbose = TRUE)
```

Arguments

genoData	An object of class GenotypeData from the package GWASTools containing the genotype data for SNPs and samples to be used for the analysis. This object can easily be created from a matrix of SNP genotype data, PLINK files, or GDS files. Alternatively, this could be an object of class SeqVarData from the package SeqVarTools containing the genotype data for the sequencing variants and samples to be used for the analysis.
pcMat	An optional matrix of principal components (PCs) to be used for ancestry adjustment. Each column represents a PC, and each row represents an individual. IDs for each individual must be set as the row names of the matrix.
freq.type	A character string taking the values 'individual' or 'population' indicating whether genotype values should be adjusted by individual-specific allele frequencies or population average allele frequencies. This should be set to 'individual' (the default) in order to do a PC-Relate analysis; see 'Details' for more information.
scale	A character string taking the values 'overall', 'variant', or 'none' indicating how genotype values should be standardized. This should be set to 'overall' (the default) in order to do a PC-Relate analysis; see 'Details' for more information.
ibd.probs	Logical indicator of whether pairwise IBD sharing probabilities (k0, k1, k2) should be estimated; the default is TRUE.
scan.include	A vector of IDs for samples to include in the analysis. If NULL, all samples in genoData are included.
training.set	An optional vector of IDs identifying which samples to use for estimation of the ancestry effect when estimating individual-specific allele frequencies. If NULL, all samples in scan.include are used. See 'Details' for more information.
scan.block.size	The number of individuals to read-in/analyze at once; the default value is 5000. See 'Details' for more information.
snp.include	A vector of SNP IDs to include in the analysis. If NULL, see chromosome for further details.

<code>chromosome</code>	A vector of integers specifying which chromosomes to analyze. This parameter is only considered when <code>snp.include</code> is NULL; if <code>chromosome</code> is also NULL, then all SNPs are included.
<code>snp.block.size</code>	The number of SNPs to read-in/analyze at once. The default value is 10000.
<code>MAF</code>	Minor allele frequency filter. When <code>freq.type</code> is 'individual', if an individual's estimated individual-specific minor allele frequency at a SNP is less than this value, that SNP will be excluded from the analysis for that individual. When <code>freq.type</code> is 'population', any SNP with a population minor allele frequency less than this value will be excluded from the analysis. The default value is 0.01.
<code>write.to.gds</code>	Logical indicator of whether the output should be written to GDS files. If FALSE (the default), then the output is returned to the R console as expected. See 'Details' for more information.
<code>gds.prefix</code>	File path specifying where to save the output when <code>write.to.gds = TRUE</code> . If NULL, the prefix 'tmp' is used. See 'Details' for more information.
<code>correct</code>	Logical indicator of whether to implement a small sample correction.
<code>verbose</code>	Logical indicator of whether updates from the function should be printed to the console; the default is TRUE.

Details

The basic premise of PC-Relate is to estimate kinship coefficients, IBD sharing probabilities, and inbreeding coefficients that reflect recent family (pedigree) relatedness by conditioning out genetic similarity due to distant population structure (ancestry) with ancestry representative principal components (PCs).

It is important that the PCs used in `pcMat` to adjust for ancestry are representative of ancestry and NOT family structure, so we recommend using PCs calculated with PC-AiR.

It is important that the order of individuals in the matrix `pcMat` matches the order of individuals in `genoData`.

In order to perform relatedness estimation, allele frequency estimates are required for centering and scaling genotype values. When `freq.type` is 'individual', individual-specific allele frequencies calculated for each individual at each SNP using the PCs specified in `pcMat` are used. When `freq.type` is 'population', population average allele frequencies calculated at each SNP are used for all individuals. (Note that when `freq.type` is set to 'population' there is no ancestry adjustment and the relatedness estimates will be confounded with population structure (ancestry)). There are multiple choices for how genotype values are scaled. When `scale` is 'variant', centered genotype values at each SNP are divided by their expected variance under Hardy-Weinberg equilibrium. When `scale` is 'overall', centered genotype values at all SNPs are divided by the average across all SNPs of their expected variances under Hardy-Weinberg equilibrium; this scaling leads to more stable behavior when using low frequency variants. When `scale` is 'none', genotype values are only centered and not scaled; this won't provide accurate kinship coefficient estimates but may be useful for other purposes. At a particular SNP, the variance used for scaling is either calculated separately for each individual using their individual-specific allele frequencies (when `freq.type` is 'individual') or once for all individuals using the population average allele frequency (when `freq.type` is 'population'). Set `freq.type` to 'individual' and `scale` to 'overall' to perform a standard PC-Relate analysis; these are the defaults. If `freq.type` is set to 'individual' and `scale` is set to 'variant', the estimators are very similar to REAP. If `freq.type` is set to 'population' and `scale` is set to 'variant', the estimators are very similar to EIGENSOFT.

The optional input `training.set` allows the user to specify which samples are used to estimate the ancestry effect when estimating individual-specific allele frequencies (if `freq.type` is 'individual') or to estimate the population allele frequency (if `freq.type` is 'population'). Ideally, `training.set`

is a set of mutually unrelated individuals. If prior information regarding pedigree structure is available, this can be used to select `training.set`, or if `pcair` was used to obtain the PCs, then the individuals in the PC-AiR 'unrelated subset' can be used. If no prior information is available, all individuals should be used.

The `scan.block.size` can be specified to alleviate memory issues when working with very large data sets. If `scan.block.size` is smaller than the number of individuals included in the analysis, then individuals will be analyzed in separate blocks. This reduces the memory required for the analysis, but genotype data must be read in multiple times for each block (to analyze all pairs), which increases the number of computations required. NOTE: if individuals are broken up into more than 1 block, `write.to.gds` must be TRUE (see below).

If `write.to.gds = TRUE`, then the output is written to two GDS files rather than returned to the R console. Use of this option requires the `gdsfmt` package. The first GDS file, named "`<gds.prefix>_freq.gds`", contains the individual-specific allele frequency estimates for each individual at each SNP (when `freq.type` is 'individual') or the population allele frequency estimates at each SNP (when `freq.type` is 'population'). The second GDS file, named "`<gds.prefix>_pcrelate.gds`", contains the PC-Relate output as described in Value below.

Value

An object of class 'pcrelate'. A list including:

<code>sample.id</code>	A vector of IDs for samples included in the analysis.
<code>kinship</code>	A matrix of estimated pairwise kinship coefficients. The order of samples matches <code>sample.id</code> .
<code>ibd.probs</code>	A matrix of estimated pairwise IBD sharing probabilities; the lower triangle gives <code>k0</code> (the probability of sharing 0 alleles IBD), the upper triangle gives <code>k2</code> (the probability of sharing 2 alleles IBD), and the diagonal is missing. The order of samples matches <code>sample.id</code> . This matrix is returned only if <code>ibd.probs = TRUE</code> in the input.
<code>nsnp</code>	A matrix specifying the the number of SNPs used to estimate the relatedness measures for each pair of individuals. The order of samples matches <code>sample.id</code> .
<code>kincorrect</code>	A vector specifying the correction factors used for the small sample correction, or NULL.
<code>k2correct</code>	A vector specifying the correction factors used for the small sample correction, or NULL.
<code>call</code>	The function call passed to <code>pcrelate</code> .
<code>method</code>	A character string. Either 'PC-Relate' or 'Unadjusted' identifying which method was used for computing relatedness estimates. 'Unadjusted' is used when <code>pcMat = NULL</code> and corresponds to an assumption of population homogeneity.

Note

The `GenotypeData` function in the `GWASTools` package should be used to create the input `genoData`. Input to the `GenotypeData` function can easily be created from an R matrix or GDS file. PLINK `.bed`, `.bim`, and `.fam` files can easily be converted to a GDS file with the function `snpgdsBED2GDS` in the `SNPRelate` package. Alternatively, the `SeqVarData` function in the `SeqVarTools` package can be used to create the input `genodata` when working with sequencing data.

Author(s)

Matthew P. Conomos

References

- Conomos M.P., Reiner A.P., Weir B.S., & Thornton T.A. (2016). Model-free Estimation of Recent Genetic Relatedness. *American Journal of Human Genetics*, 98(1), 127-148.
- Gogarten, S.M., Bhangale, T., Conomos, M.P., Laurie, C.A., McHugh, C.P., Painter, I., ... & Laurie, C.C. (2012). GWASTools: an R/Bioconductor package for quality control and analysis of Genome-Wide Association Studies. *Bioinformatics*, 28(24), 3329-3331.

See Also

[pcrelateReadKinship](#), [pcrelateReadInbreed](#), and [pcrelateMakeGRM](#) for functions that can be used to read in the results output by [pcrelate](#). [GWASTools](#) for a description of the package containing the following functions: [GenotypeData](#) for a description of creating a GenotypeData class object for storing sample and SNP genotype data, [MatrixGenotypeReader](#) for a description of reading in genotype data stored as a matrix, and [GdsGenotypeReader](#) for a description of reading in genotype data stored as a GDS file. Also see [snpgdsBED2GDS](#) in the [SNPRelate](#) package for a description of converting binary PLINK files to GDS.

Examples

```
library(GWASTools)

# file path to GDS file
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
# read in GDS data
HapMap_geno <- GdsGenotypeReader(filename = gdsfile)
# create a GenotypeData class object
HapMap_genoData <- GenotypeData(HapMap_geno)
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")
# run PC-AiR
mypcair <- pcair(genoData = HapMap_genoData, kinMat = HapMap_ASW_MXL_KINGmat,
                divMat = HapMap_ASW_MXL_KINGmat)
# run PC-Relate
mypcrel <- pcrelate(genoData = HapMap_genoData, pcMat = mypcair$vectors[,1],
                  training.set = mypcair$unrels)
close(HapMap_genoData)
```

pcrelateMakeGRM	<i>Creates a Genetic Relationship Matrix (GRM) of Pairwise Kinship Coefficient Estimates from PC-Relate Output</i>
-----------------	--

Description

[pcrelateMakeGRM](#) is used to create a genetic relationship matrix (GRM) of pairwise kinship coefficient estimates from the output of [pcrelate](#).

Usage

```
pcrelateMakeGRM(pcrelObj, scan.include = NULL, scaleKin = 2)
```


Arguments

pcrelObj	The object containing the output from pcrelate. This could be a list of class pcrelate or an object of class gds.class read into R using the function openfn.gds from the gdsfmt package.
scan.include	A vector of IDs for samples to be included in the GRM. The default is NULL, which includes all samples in pcrelObj.
scaleKin	Specifies a numeric constant to scale each estimated kinship coefficient by in the GRM. The default value is 2.

Details

This function provides a quick and easy way to construct a genetic relationship matrix (GRM) from the output of pcrelate.

Author(s)

Matthew P. Conomos

References

Conomos M.P., Reiner A.P., Weir B.S., & Thornton T.A. (2016). Model-free Estimation of Recent Genetic Relatedness. *American Journal of Human Genetics*, 98(1), 127-148.

See Also

[pcrelate](#) for the function that performs PC-Relate. [pcrelateReadKinship](#) for the function that creates a table of pairwise kinship coefficient and IBD sharing probabilities from the same PC-Relate output file. [pcrelateReadInbreed](#) for the function that creates a table of inbreeding coefficient estimates from the same PC-Relate output file.

pcrelateReadInbreed *Create a Table of Inbreeding Coefficient Estimates from PC-Relate Output*

Description

pcrelateReadInbreed is used to create a table of inbreeding coefficient estimates from the output of pcrelate.

Usage

```
pcrelateReadInbreed(pcrelObj, scan.include = NULL, f.thresh = NULL)
```

Arguments

pcrelObj	The object containing the output from pcrelate. This could be a list of class pcrelate or an object of class gds.class read into R using the function openfn.gds from the gdsfmt package.
scan.include	A vector of IDs for samples to be included in the table. The default is NULL, which includes all samples in pcrelObj.

`f.thresh` Specifies a minimum value of the estimated inbreeding coefficient to include in the table; i.e. only individuals with an estimated inbreeding coefficient greater than `f.thresh` will be included in the table. The default is `NULL`, which includes all individuals.

Details

This function provides an easy way to make a table of estimated inbreeding coefficients.

Author(s)

Matthew P. Conomos

References

Conomos M.P., Reiner A.P., Weir B.S., & Thornton T.A. (2016). Model-free Estimation of Recent Genetic Relatedness. *American Journal of Human Genetics*, 98(1), 127-148.

See Also

[pcrelate](#) for the function that performs PC-Relate. [pcrelateReadKinship](#) for the function that creates a table of pairwise kinship coefficient and IBD sharing probabilities from the same PC-Relate output file. [pcrelateMakeGRM](#) for the function that creates a genetic relationship matrix (GRM) of pairwise kinship coefficient estimates from the same PC-Relate output file.

`pcrelateReadKinship` *Create a Table of Pairwise Kinship Coefficient and IBD Sharing Probability Estimates from PC-Relate Output*

Description

`pcrelateReadKinship` is used to create a table of pairwise kinship coefficient and IBD sharing probability (`k0`, `k1`, `k2`) estimates from the output of `pcrelate`.

Usage

```
pcrelateReadKinship(pcrelObj, scan.include = NULL, ibd.probs = TRUE,
  kin.thresh = NULL)
```

Arguments

`pcrelObj` The object containing the output from `pcrelate`. This could be a list of class `pcrelate` or an object of class `gds.class` read into R using the function `openfn.gds` from the `gdsfmt` package.

`scan.include` A vector of IDs for samples to be included in the table. The default is `NULL`, which includes all samples in `pcrelObj`.

`ibd.probs` Logical indicator of whether or not the output in `pcrelObj` has estimates of IBD sharing probabilities.

`kin.thresh` Specifies a minimum value of the estimated kinship coefficient to include in the table; i.e. only pairs with an estimated kinship coefficient greater than `kin.thresh` will be included in the table. The default is `NULL`, which includes all pairs.

Details

This function provides an easy way to make a table of pairwise relatedness estimates.

Author(s)

Matthew P. Conomos

References

Conomos M.P., Reiner A.P., Weir B.S., & Thornton T.A. (2016). Model-free Estimation of Recent Genetic Relatedness. *American Journal of Human Genetics*, 98(1), 127-148.

See Also

[pcrelate](#) for the function that performs PC-Relate. [pcrelateReadInbreed](#) for the function that creates a table of inbreeding coefficient estimates from the same PC-Relate output file. [pcrelateMakeGRM](#) for the function that creates a genetic relationship matrix (GRM) of pairwise kinship coefficient estimates from the same PC-Relate output file.

plot.pcair

PC-AiR: Plotting PCs

Description

plot.pcair is used to plot pairs of principal components contained in a class 'pcair' object obtained as output from the pcair function.

Usage

```
## S3 method for class 'pcair'
plot(x, vx = 1, vy = 2, pch = NULL, col = NULL,
      xlim = NULL, ylim = NULL, main = NULL, sub = NULL,
      xlab = NULL, ylab = NULL, ...)
```

Arguments

x	An object of class 'pcair' obtained as output from the pcair function.
vx	An integer indicating which principal component to plot on the x-axis; the default is 1.
vy	An integer indicating which principal component to plot on the y-axis; the default is 2.
pch	Either an integer specifying a symbol or a single character to be used in plotting points. If NULL, the default is dots for the 'unrelated subset' and + for the 'related subset'.
col	A specification for the plotting color for points. If NULL, the default is black for the 'unrelated subset' and blue for the 'related subset'.
xlim	The range of values shown on the x-axis. If NULL, the default shows all points.
ylim	The range of values shown on the y-axis. If NULL, the default shows all points.

main	An overall title for the plot. If NULL, the default specifies which PC-AiR PCs are plotted.
sub	A sub title for the plot. If NULL, the default is none.
xlab	A title for the x-axis. If NULL, the default specifies which PC-AiR PC is plotted.
ylab	A title for the y-axis. If NULL, the default specifies which PC-AiR PC is plotted.
...	Other parameters to be passed through to plotting functions, (see par).

Details

This function provides a quick and easy way to plot principal components obtained with the function `pcair` to visualize the population structure captured by PC-AiR.

Value

A figure showing the selected principal components plotted against each other.

Author(s)

Matthew P. Conomos

See Also

[pcair](#) for obtaining principal components that capture population structure in the presence of relatedness. [par](#) for more in depth descriptions of plotting parameters. The generic function [plot](#).

Examples

```
library(GWASTools)

# file path to GDS file
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
# read in GDS data
HapMap_geno <- GdsGenotypeReader(filename = gdsfile)
# create a GenotypeData class object
HapMap_genoData <- GenotypeData(HapMap_geno)
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")
# run PC-AiR
mypcair <- pcair(genoData = HapMap_genoData, kinMat = HapMap_ASW_MXL_KINGmat,
                divMat = HapMap_ASW_MXL_KINGmat)
# plot top 2 PCs
plot(mypcair)
# plot PCs 3 and 4
plot(mypcair, vx = 3, vy = 4)
close(HapMap_genoData)
```

`varCompCI`*Variance Component Confidence Intervals*

Description

`varCompCI` provides confidence intervals for the variance component estimates found using `fitNullMM`. The confidence intervals can be found on either the original scale or for the proportion of total variability explained.

Usage

```
varCompCI(nullMMobj, prop = TRUE)
```

Arguments

<code>nullMMobj</code>	A null model object returned by <code>fitNullMM</code> .
<code>prop</code>	A logical indicator of whether the point estimates and confidence intervals should be returned as the proportion of total variability explained (TRUE) or on the original scale (FALSE).

Details

`varCompCI` takes the object returned by `fitNullMM` as its input and returns point estimates and confidence intervals for each of the random effects variance component estimates. If a kinship matrix or genetic relationship matrix (GRM) was included as a random effect in the model fit using `fitNullMM`, then this function can be used to provide a heritability estimate when `prop` is TRUE.

Value

`varCompCI` prints a table of point estimates and 95% confidence interval limits for each estimated variance component.

Author(s)

Matthew P. Conomos

See Also

`fitNullMM` for fitting the mixed model and performing the variance component estimation.

Examples

```
library(GWASTools)

# file path to GDS file
gdsfile <- system.file("extdata", "HapMap_ASW_MXL_geno.gds", package="GENESIS")
# read in GDS data
HapMap_geno <- GdsGenotypeReader(filename = gdsfile)
# create a GenotypeData class object
HapMap_genoData <- GenotypeData(HapMap_geno)
# load saved matrix of KING-robust estimates
data("HapMap_ASW_MXL_KINGmat")
```

```
# run PC-AiR
mypcair <- pcair(genoData = HapMap_genoData, kinMat = HapMap_ASW_MXL_KINGmat,
                divMat = HapMap_ASW_MXL_KINGmat)

# run PC-Relate
mypcrel <- pcrelate(genoData = HapMap_genoData, pcMat = mypcair$vector[,1],
                  training.set = mypcair$unrels)
close(HapMap_genoData)

# generate a phenotype
set.seed(4)
pheno <- 0.2*mypcair$vector[,1] + rnorm(mypcair$nsamp, mean = 0, sd = 1)

# make ScanAnnotationDataFrame
scanAnnot <- ScanAnnotationDataFrame(data.frame(scanID = mypcrel$sample.id,
                                              pc1 = mypcair$vector[,1], pheno = pheno))

# make covMatList
covMatList <- list("Kin" = pcrelateMakeGRM(mypcrel))

# fit the null mixed model
nullmod <- fitNullMM(scanData = scanAnnot, outcome = "pheno", covars = "pc1", covMatList = covMatList)

# find the variance component CIs
varCompCI(nullmod, prop = TRUE)
varCompCI(nullmod, prop = FALSE)
```

Index

- *Topic **ancestry**
 - [pcair](#), [23](#)
 - *Topic **association**
 - [admixMapMM](#), [3](#)
 - [assocTestMM](#), [5](#)
 - [assocTestSeq](#), [9](#)
 - [assocTestSeqWindow](#), [12](#)
 - [fitNullMM](#), [16](#)
 - [fitNullReg](#), [20](#)
 - *Topic **datasets**
 - [HapMap_ASW_MXL_KINGmat](#), [21](#)
 - *Topic **heritability**
 - [varCompCI](#), [37](#)
 - *Topic **mixed model**
 - [admixMapMM](#), [3](#)
 - [assocTestMM](#), [5](#)
 - [fitNullMM](#), [16](#)
 - [varCompCI](#), [37](#)
 - *Topic **multivariate**
 - [pcair](#), [23](#)
 - *Topic **relatedness**
 - [pcrelate](#), [29](#)
 - *Topic **robust**
 - [pcair](#), [23](#)
 - [pcrelate](#), [29](#)
 - *Topic **variance component**
 - [fitNullMM](#), [16](#)
 - [varCompCI](#), [37](#)
-
- [admixMapMM](#), [3](#)
 - [assocTestMM](#), [3](#), [4](#), [5](#), [16](#), [19](#)
 - [assocTestSeq](#), [9](#), [20](#)
 - [assocTestSeqWindow](#), [12](#), [20](#)
-
- [family](#), [17](#), [21](#)
 - [fitNullMM](#), [3–5](#), [8](#), [9](#), [13](#), [16](#), [37](#)
 - [fitNullReg](#), [9](#), [13](#), [20](#)
-
- [gdsfmt](#), [31](#), [33](#), [34](#)
 - [GdsGenotypeReader](#), [8](#), [26](#), [32](#)
 - [GENESIS \(GENESIS-package\)](#), [2](#)
 - [GENESIS-package](#), [2](#)
 - [GenotypeData](#), [4](#), [8](#), [26](#), [32](#)
 - [GWASTools](#), [8](#), [19](#), [26](#), [32](#)
 - [HapMap_ASW_MXL_KINGmat](#), [21](#)
 - [king2mat](#), [3](#), [22](#), [26](#), [28](#)
 - [manhattanPlot](#), [8](#)
 - [MatrixGenotypeReader](#), [8](#), [26](#), [32](#)
 - [openfn.gds](#), [33](#), [34](#)
 - [par](#), [36](#)
 - [pcair](#), [3](#), [22](#), [23](#), [23](#), [28](#), [31](#), [36](#)
 - [pcairPartition](#), [3](#), [22](#), [23](#), [26](#), [27](#)
 - [pcrelate](#), [3](#), [29](#), [33–35](#)
 - [pcrelateMakeGRM](#), [3](#), [32](#), [32](#), [34](#), [35](#)
 - [pcrelateReadInbreed](#), [3](#), [32](#), [33](#), [33](#), [35](#)
 - [pcrelateReadKinship](#), [3](#), [32–34](#), [34](#)
 - [plot](#), [36](#)
 - [plot.pcair](#), [3](#), [26](#), [35](#)
 - [print](#), [26](#)
 - [print.pcair \(pcair\)](#), [23](#)
 - [print.summary.pcair \(pcair\)](#), [23](#)
 - [qqPlot](#), [8](#)
 - [ScanAnnotationDataFrame](#), [19](#)
 - [snpgdsBED2GDS](#), [8](#), [26](#), [32](#)
 - [SNPRelate](#), [8](#), [26](#), [32](#)
 - [summary](#), [26](#)
 - [summary.pcair \(pcair\)](#), [23](#)
 - [varCompCI](#), [3](#), [19](#), [37](#)