Package 'AllelicSeries'

April 2, 2025

Title Allelic Series Test

Version 0.1.1.5

Description Implementation of gene-level rare variant association tests targeting allelic series: genes where increasingly deleterious mutations have increasingly large phenotypic effects. The COding-variant Allelic Series Test (COAST) operates on the benign missense variants (BMVs), deleterious missense variants (DMVs), and protein truncating variants (PTVs) within a gene. COAST uses a set of adjustable weights that tailor the test towards rejecting the null hypothesis for genes where the average magnitude of effect increases monotonically from BMVs to DMVs to PTVs. See McCaw ZR, O'Dushlaine C, Somineni H, Bereket M, Klein C, Karaletsos T, Casale FP, Koller D, Soare TW. (2023) ``An allelic series rare variant association test for candidate gene discovery'' <doi:10.1016/j.ajhg.2023.07.001>.

License BSD_3_clause + file LICENSE

URL https://github.com/insitro/AllelicSeries

Encoding UTF-8

Imports CompQuadForm, glue, methods, Rcpp, RNOmni, SKAT

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation yes

Author Zachary McCaw [aut, cre] (<https://orcid.org/0000-0002-2006-9828>), Christoph Klein [ctb] (<https://orcid.org/0000-0002-1748-625X>), Thomas Soare [ctb] (<https://orcid.org/0000-0002-4903-8646>), Jianhui Gao [ctb] (<https://orcid.org/0000-0003-0915-1473>), insitro [cph]

Maintainer Zachary McCaw <zmccaw@alumni.harvard.edu>

Repository CRAN

Date/Publication 2025-04-02 21:50:13 UTC

Contents

`Aggregator` *Aggregator*

Description

Aggregates genotypes within annotation categories.

Usage

```
Aggregator(
  anno,
  geno,
  drop_empty = TRUE,
  indicator = FALSE,
  method = "none",
  min_mac = 0,
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
geno	(n x snps) genotype matrix.
drop_empty	Drop empty columns? Default: TRUE.
indicator	Convert raw counts to indicators? Default: FALSE.
method	Method for aggregating across categories: ("none", "max", "sum"). Default: "none".
min_mac	Minimum minor allele count for inclusion. Default: 0.
weights	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.

Value

(n x L) Numeric matrix without weighting, (n x 1) numeric matrix with weighting.

Notes

- Ensure the length of the `weights` vector matches the total number of annotation categories.
- The `weights` essentially scales the minor allele count in the `l`th category by `weights[l]`.

ASBT *Allelic Series Burden Test*

Description

Burden test with allelic series weights.

Usage

```
ASBT(
  anno,
  geno,
  pheno,
  apply_int = TRUE,
  covar = NULL,
  indicator = FALSE,
  is_pheno_binary = FALSE,
  method = "none",
  min_mac = 0,
  return_beta = FALSE,
  score_test = FALSE,
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
geno	(n x snps) genotype matrix.
pheno	(n x 1) phenotype vector.
apply_int	Apply rank-based inverse normal transform to the phenotype? Default: TRUE. Ignored if phenotype is binary.
covar	(n x p) covariate matrix. Defaults to an (n x 1) intercept.
indicator	Convert raw counts to indicators?
is_pheno_binary	Is the phenotype binary? Default: FALSE.
method	Method for aggregating across categories: ("none", "max", "sum"). Default: "none".
min_mac	Minimum minor allele count for inclusion. Default: 0.
return_beta	Return the estimated effect size? Default: FALSE.
score_test	Run a score test? If FALSE, performs a Wald test.
weights	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.

Value

If `return_beta = TRUE`, a list of including the effect size data.frame "betas" and the p-value "pval". If `return_beta = FALSE`, a numeric p-value.

Examples

```
# Generate data.
data <- DGP(n = 1e3, snps = 1e2)
```

```
# Run the Allelic Series Burden Test.
# Note: the output is a scalar p-value.
results <- ASBT(
  anno = data$anno,
  geno = data$geno,
  pheno = data$pheno,
  covar = data$covar
)
```

ASBTSS *Allelic Series Burden Test from Summary Statistics*

Description

Allelic series burden test from summary statistics.

Usage

```
ASBTSS(
  anno,
  beta,
  se,
  check = TRUE,
  eps = 1,
  lambda = 1,
  ld = NULL,
  method = "none",
  return_beta = FALSE,
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
beta	(snps x 1) vector of effect sizes for the coding genetic variants within a gene.
se	(snps x 1) vector of standard errors for the effect sizes.
check	Run input checks? Default: TRUE.
eps	Epsilon added to the diagonal of the LD matrix if not positive definite. Note, smaller values increase the chances of a false positive.
lambda	Optional genomic inflation factor. Defaults to 1, which results in no rescaling.
ld	(snps x snps) matrix of correlations among the genetic variants. Although ideally provided, an identity matrix is assumed if not.
method	Method for aggregating across categories: ("none", "sum"). Default: "none".
return_beta	Return the estimated effect size? Default: FALSE.
weights	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.

Value

If `return_beta = TRUE`, a list of including the effect size data.frame "betas" and the p-value "pval". If `return_beta = FALSE`, a numeric p-value.

Notes

- The allelic series burden does not require the minor allele frequencies.

Examples

```
# Generate data.
data <- DGP(n = 1e3)
sumstats <- CalcSumstats(data = data)

# Run allelic series burden test from sumstats.
results <- ASBTSS(
  anno = sumstats$sumstats$anno,
  beta = sumstats$sumstats$beta,
  se = sumstats$sumstats$se,
  ld = sumstats$ld
)
show(results)
```

ASKAT *Allelic Series SKAT Test*

Description

Sequence kernel association test (SKAT) with allelic series weights.

Usage

```
ASKAT(
  anno,
  geno,
  pheno,
  apply_int = TRUE,
  covar = NULL,
  is_pheno_binary = FALSE,
  min_mac = 0,
  return_null_model = FALSE,
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
geno	(n x snps) genotype matrix.
pheno	(n x 1) phenotype vector.
apply_int	Apply rank-based inverse normal transform to the phenotype? Default: TRUE. Ignored if phenotype is binary.
covar	(n x p) covariate matrix. Defaults to an (n x 1) intercept.
is_pheno_binary	Is the phenotype binary? Default: FALSE.
min_mac	Minimum minor allele count for inclusion. Default: 0.
return_null_model	Return the null model in addition to the p-value? Useful if running additional SKAT tests. Default: FALSE.
weights	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.

Value

If `return_null_model`, a list containing the p-value and the SKAT null model. Otherwise, a numeric p-value.

Examples

```
# Generate data.
data <- DGP(n = 1e3, snps = 1e2)

# Run the Allelic Series SKAT Test.
# Note: the output is a scalar p-value.
results <- ASKAT(
  anno = data$anno,
  geno = data$geno,
  pheno = data$pheno,
  covar = data$covar
)
```

ASKATSS *Allelic Series SKAT-O from Summary Statistics*

Description

Allelic series sequence kernel association test from summary statistics.

Usage

```
ASKATSS(
  anno,
  beta,
  se,
  check = TRUE,
  eps = 1,
  lambda = 1,
  ld = NULL,
  maf = NULL,
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
beta	(snps x 1) vector of effect sizes for the coding genetic variants within a gene.
se	(snps x 1) vector of standard errors for the effect sizes.
check	Run input checks? Default: TRUE.
eps	Epsilon added to the diagonal of the LD matrix if not positive definite. Note, smaller values increase the chances of a false positive.
lambda	Optional genomic inflation factor. Defaults to 1, which results in no rescaling.
ld	(snps x snps) matrix of correlations among the genetic variants. Although ideally provided, an identity matrix is assumed if not.
maf	(snps x 1) vector of minor allele frequencies. Although ideally provided, defaults to the zero vector.
weights	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.

Value

Numeric p-value of the allelic series SKAT-O test.

Notes

- The SKAT test requires per-variant minor allele frequencies (MAFs) for the purpose of upweighting rarer variants. If unknown, `maf` can be safely omitted. The only consequence is that the SKAT weights will no longer be inversely proportional to the genotypic variance.

Examples

```
# Generate data.
data <- DGP(n = 1e3)
sumstats <- CalcSumstats(data = data)

# Run allelic series SKAT test from sumstats.
```

```
# Note: the SKAT test requires MAF.
results <- ASKATSS(
  anno = sumstats$sumstats$anno,
  beta = sumstats$sumstats$beta,
  maf = sumstats$sumstats$maf,
  se = sumstats$sumstats$se,
  ld = sumstats$ld
)
show(results)
```

`BaselineSS` *Baseline Counts Test from Sumstats*

Description

Baseline Counts Test from Sumstats

Usage

```
BaselineSS(anno, beta, ld, se, n_anno = 3L, return_beta = FALSE)
```

Arguments

`anno`	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
`beta`	(snps x 1) vector of effect sizes for the coding genetic variants within a gene.
`ld`	(snps x snps) matrix of correlations among the genetic variants.
`se`	(snps x 1) vector of standard errors for the effect sizes.
`n_anno`	Number of annotation categories L.
`return_beta`	Return estimated effect sizes and standard errors? Default: FALSE.

Value

If `return_beta`, a list containing the category effect sizes, standard errors, and the p-value. Otherwise, the numeric p-value only.

CalcRegParam *Calculate Regression Parameters*

Description

Calculate phenotypic regression coefficients and the residual variation based on proportion of variation explained (PVE) by each factor.

Usage

```
CalcRegParam(pve_age = 0.1, pve_pcs = 0.2, pve_sex = 0.1)
```

Arguments

pve_age	PVE by age.
pve_pcs	PVE by PCs (collectively).
pve_sex	PVE by sex.

Value

List containing the (5 x 1) regression coefficient vector "coef" and the residual standard deviation "sd".

CalcSumstats *Calculate Summary Statistics*

Description

Generate summary statistics from individual-level data. Provide either a list of data as generated by DGP, or all of anno, geno, and pheno.

Usage

```
CalcSumstats(
  anno = NULL,
  covar = NULL,
  data = NULL,
  geno = NULL,
  pheno = NULL,
  add_intercept = TRUE,
  is_binary = FALSE
)
```

Arguments

anno	(snps x 1) annotation vector.
covar	(subjects x covars) covariate matrix.
data	List of data containing the annotation vector anno, the covariate data covar, the genotype matrix geno, and the phenotype vector pheno, as returned by DGP. Overrides the other arguments if provided.
geno	(subjects x snps) genotype matrix.
pheno	(subjects x 1) phenotype vector.
add_intercept	Add an intercept if not present in the supplied covariate matrix covar? Default: TRUE.
is_binary	Is the phenotype binary? Default: FALSE.

Value

List containing the following items:

- ld: A SNP x SNP correlation (LD) matrix.
- sumstats: A SNP x 5 matrix of summary statistics, including the . annotation, MAF, estimated effect size, standard error, and p-value.
- type: Either "binary" or "quantitative".‘

Examples

```
data <- DGP()
sumstats <- CalcSumstats(data = data)
```

CheckInputs *Check Inputs*

Description

Check Inputs

Usage

```
CheckInputs(anno, covar, geno, is_pheno_binary, pheno, weights)
```

Arguments

anno	(snps x 1) annotation vector.
covar	(n x p) covariate matrix.
geno	(n x snps) genotype matrix.
is_pheno_binary	Is the phenotype binary?
pheno	(n x 1) phenotype vector.
weights	(L x 1) annotation category weights.

Value

None.

`CheckInputsSS` *Input Checks for Summary Statistics*

Description

Input Checks for Summary Statistics

Usage

```
CheckInputsSS(anno, beta, se, lambda, ld, weights, is_skat = FALSE, maf = NULL)
```

Arguments

`anno`	(snps x 1) annotation vector with values in c(0, 1, 2).
`beta`	(snps x 1) vector of effect sizes for the coding genetic variants within a gene.
`se`	(snps x 1) vector of standard errors for the effect sizes.
`lambda`	Genomic inflation factor.
`ld`	(snps x snps) matrix of correlations among the genetic variants. Although ideally provided, an identity matrix is assumed if not.
`weights`	(L x 1) annotation category weights.
`is_skat`	Logical, is the check for the SKAT test?
`maf`	(snps x 1) vector of minor allele frequencies. Although ideally provided, defaults to the zero vector.

Value

Logical indicating whether the matrix was positive definite.

`COAST` *COding-variant Allelic Series Test*

Description

Main allelic series test. Performs both Burden and SKAT type tests, then combines the results to calculate an omnibus p-value.

Usage

```
COAST(
  anno,
  geno,
  pheno,
  add_intercept = TRUE,
  apply_int = TRUE,
  covar = NULL,
  include_orig_skato_all = FALSE,
  include_orig_skato_ptv = FALSE,
  is_pheno_binary = FALSE,
  min_mac = 0,
  ptv_anno = 3,
  pval_weights = NULL,
  return_omni_only = FALSE,
  score_test = FALSE,
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
geno	(n x snps) genotype matrix.
pheno	(n x 1) phenotype vector.
add_intercept	Add an intercept if not present in the supplied covariate matrix covar? Default: TRUE.
apply_int	Apply rank-based inverse normal transform to the phenotype? Default: TRUE. Ignored if phenotype is binary.
covar	(n x p) covariate matrix. Defaults to an (n x 1) intercept.
include_orig_skato_all	Include the original version of SKAT-O applied to all variants in the omnibus test? Default: FALSE.
include_orig_skato_ptv	Include the original version of SKAT-O applied to PTV variants only in the omnibus test? Default: FALSE.
is_pheno_binary	Is the phenotype binary? Default: FALSE.
min_mac	Minimum minor allele count for inclusion. Default: 0.
ptv_anno	Annotation of the PTV category, only required if include_orig_skato_ptv is set to TRUE.
pval_weights	Optional vector of relative weights for combining the component tests to perform the omnibus test. By default, 50% of weight is given to the 6 burden tests, and 50% to the 1 SKAT test. If specified, the weight vector should have length 7, and the length should be increased if either include_orig_skato_all or include_orig_skato_ptv is active.

return_omni_only
: Return only the omnibus p-value? Default: FALSE.

score_test
: Use a score test for burden analysis? If FALSE, uses a Wald test.

weights
: (L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.

Value

An object of class `COAST` with slots for effect sizes, variant counts, and p-values.

Examples

```
# Generate data.
data <- DGP(n = 1e3, snps = 1e2)

# Run the COding-variant Allelic Series Test.
results <- COAST(
  anno = data$anno,
  geno = data$geno,
  pheno = data$pheno,
  covar = data$covar
)
show(results)
```

COAST-class *Allelic Series Output Class*

Description

Allelic Series Output Class

Slots

`Betas` Effect sizes and standard errors.

`Counts` Allele, variant, and carrier counts.

`Pvals` Result p-values.

COASTSS *COding-variant Allelic Series Test from Summary Statistics*

Description

Main function for performing the allelic series test from summary statistics. Performs both Burden and SKAT type tests, then combines the results to calculate an omnibus p-value. Note that not all tests included in COAST are available when working with summary statistics.

Usage

```
COASTSS(
  anno,
  beta,
  se,
  check = TRUE,
  eps = 1,
  lambda = c(1, 1, 1),
  ld = NULL,
  maf = NULL,
  pval_weights = c(0.25, 0.25, 0.5),
  weights = c(1, 2, 3)
)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
beta	(snps x 1) vector of effect sizes for the coding genetic variants within a gene.
se	(snps x 1) vector of standard errors for the effect sizes.
check	Run input checks? Default: TRUE.
eps	Epsilon added to the diagonal of the LD matrix if not positive definite. Note, epsilon should increase as the sample size decreases.
lambda	Optional (3 x 1) vector of inflation factors, one for each component test. Defaults to a 1s vector, which results in no rescaling.
ld	(snps x snps) matrix of correlations among the genetic variants. Although ideally provided, an identity matrix is assumed if not.
maf	(snps x 1) vector of minor allele frequencies. Although ideally provided, defaults to the zero vector.
pval_weights	(3 x 1) vector of relative weights for combining the component tests to perform the omnibus test. The default of c(0.25, 0.25, 0.50) gives the SKAT test equal weight to the two burden tests.
weights	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of weights.

Value

Numeric p-value.

Examples

```
# Generate data.
data <- DGP(n = 1e3)
sumstats <- CalcSumstats(data = data)

# Run the Coding-variant Allelic Series Test from summary statistics.
results <- COASTSS(
  anno = sumstats$sumstats$anno,
  beta = sumstats$sumstats$beta,
  se = sumstats$sumstats$se,
  ld = sumstats$ld,
  maf = sumstats$sumstats$maf,
)
show(results)
```

CollapseGeno *Collapse Variants*

Description

Collapse variants with minor allele counts below the `min_mac` threshold into an aggregated variant, separately within each variant category. Note that the ordering of the variants will change, and that collapsing does not guarantee that the resulting aggregate variant will itself have a MAC greater than or equal to `min_mac`.

Usage

```
CollapseGeno(anno, geno, min_mac = 11)
```

Arguments

`anno`	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
`geno`	(n x snps) genotype matrix.
`min_mac`	Minimum minor allele count (MAC) for retention as an individual variant. Variants with a MAC strictly less than the minimum MAC will be collapsed into an aggregated variant, separately within each annotation category.

Value

List containing the collapsed genotypes `geno` and corresponding annotations `anno`, plus a data.frame `vars` specifying which variants were collapsed within each annotation category.

`Comparator` *Comparator Test*

Description

Runs burden, SKAT, and SKAT-O, using default settings.

Usage

```
Comparator(covar, geno, pheno, apply_int = TRUE, is_pheno_binary = FALSE)
```

Arguments

`covar`	(n x p) covariate matrix.
`geno`	(n x snps) genotype matrix.
`pheno`	(n x 1) phenotype vector.
`apply_int`	Apply rank-based inverse normal transform to the phenotype? Default: TRUE. Ignored if phenotype is binary.
`is_pheno_binary`	Is the phenotype binary? Default: FALSE.

Value

Numeric vector of p-values.

Examples

```
# Generate data.
data <- DGP(n = 1e3, snps = 1e2)

# Run the comparators.
results <- Comparator(
  geno = data$geno,
  pheno = data$pheno,
  covar = data$covar
)
```

ContainsInt *Contains Intercept?*

Description

Check if a design matrix contains an intercept.

Usage

```
ContainsInt(x)
```

Arguments

x	Design matrix.

Value

Logical.

CorCpp *Correlation C++*

Description

Correlation C++

Usage

```
CorCpp(x)
```

Arguments

x	Numeric matrix.

Value

Numeric matrix of correlation among the columns.

Notes

Verified this function is faster that R's built-in correlation function for large genotype matrices.

Counts *Count Variants and Carriers*

Description

Count Variants and Carriers

Usage

```
Counts(anno, geno, n_anno, min_mac = 0L)
```

Arguments

anno	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
geno	(n x snps) genotype matrix.
n_anno	Number of annotation categories L.
min_mac	Minimum minor allele count for inclusion. Default: 0.

Value

Data.frame of allele, variant, and carrier counts.

DfOrNULL-class *Data.frame or Null Class*

Description

Data.frame or Null Class

DGP *Data Generating Process*

Description

Generate a data set consisting of:

- anno: (snps x 1) annotation vector.
- covar: (subjects x 6) covariate matrix.
- geno: (subjects x snps) genotype matrix.
- pheno: (subjects x 1) phenotype vector.
- type: Either "binary" or "quantitative".

Usage

```
DGP(
  anno = NULL,
  beta = c(1, 2, 3),
  binary = FALSE,
  geno = NULL,
  include_residual = TRUE,
  indicator = FALSE,
  maf_range = c(0.001, 0.005),
  method = "none",
  n = 100,
  prop_anno = c(0.5, 0.4, 0.1),
  prop_causal = 1,
  random_signs = FALSE,
  random_var = 0,
  snps = 100,
  weights = c(1, 1, 1)
)
```

Arguments

anno	Annotation vector, if providing genotypes. Should match the number of columns in geno.
beta	If method = "none", a (L x 1) coefficient with effect sizes for each annotation category. By default, there are L = 3 annotation categories corresponding to BMVs, DMVs, and PTVs. If method != "none", a scalar effect size for the allelic series burden score.
binary	Generate binary phenotype? Default: FALSE.
geno	Genotype matrix, if providing genotypes.
include_residual	Include residual? If FALSE, returns the expected value. Intended for testing.
indicator	Convert raw counts to indicators? Default: FALSE.
maf_range	Range of minor allele frequencies: c(MIN, MAX).
method	Genotype aggregation method. Default: "none".
n	Sample size.
prop_anno	Proportions of annotations in each category. Length should equal the number of annotation categories. Default of c(0.5, 0.4, 0.1) is based on the approximate empirical frequencies of BMVs, DMVs, and PTVs.
prop_causal	Proportion of variants which are causal. Default: 1.0.
random_signs	Randomize signs? FALSE for burden-type genetic architecture, TRUE for SKAT-type.
random_var	Frailty variance in the case of random signs. Default: 0.
snps	Number of SNP in the gene. Default: 100.
weights	Annotation category weights. Length should match prop_anno.

Value

List containing: genotypes, annotations, covariates, phenotypes.

Examples

```
# Generate data.
data <- DGP(n = 100)

# View components.
table(data$anno)
head(data$covar)
head(data$geno[, 1:5])
hist(data$pheno)

# Generate data with L != 3 categories.
data <- DGP(
  beta = c(1, 2, 3, 4),
  prop_anno = c(0.25, 0.25, 0.25, 0.25),
  weights = c(1, 1, 1, 1)
)
```

`FilterGenos` *Filter Noncausal Variants*

Description

Remove a random fraction of variants, which are designated non-causal.

Usage

```
FilterGenos(anno, geno, prop_causal = 1)
```

Arguments

`anno`	(snps x 1) annotation vector.
`geno`	(n x snps) genotype matrix.
`prop_causal`	Proportion of variants which are causal.

Value

List containing the (n x snps) genotype matrix "geno" and the (snps x 1) annotation vector "anno".

GenAnno *Generate Genotype Annotations*

Description

Returns a vector of length = the number of columns (SNPs) in the genotype matrix. Each SNP is categorized into one of L categories, where L is determined by the length of `prop_anno`.

Usage

```
GenAnno(snps, prop_anno = c(0.5, 0.4, 0.1))
```

Arguments

`snps`	Number of SNPs in the gene.
`prop_anno`	Proportions of annotations in each category. Length should equal the number of annotation categories. Default of c(0.5, 0.4, 0.1) is based on the approximate empirical frequencies of BMVs, DMVs, and PTVs.

Value

(snps x 1) integer vector.

GenCovar *Generate Covariates*

Description

Generate an (n x 6) covariate matrix with columns representing an intercept, age, sex, and 3 genetic PCs. Because these simulations address rare variant analysis, correlation between genotypes and the genetic PCs (based on common variants) is unnecessary.

Usage

```
GenCovar(n)
```

Arguments

`n`	Sample size.

Value

(n x 6) numeric matrix.

GenGeno *Generate Genotypes*

Description

Generates genotypes in linkage equilibrium with accompanying annotations.

Usage

```
GenGeno(n, snps, maf_range = c(0.001, 0.005), prop_anno = c(0.5, 0.4, 0.1))
```

Arguments

n	Sample size.
snps	Number of SNP in the gene.
maf_range	Range of minor allele frequencies: c(MIN, MAX).
prop_anno	Proportions of annotations in each category. Length should equal the number of annotation categories. Default of c(0.5, 0.4, 0.1) is based on the approximate empirical frequencies of BMVs, DMVs, and PTVs.

Value

List containing the (n x snps) genotype matrix "geno" and the (snps x 1) annotation vector "anno".

GenGenoMat *Generate Genotype Matrix*

Description

Generate genotypes for n subject at snps variants in linkage equilibrium. Genotypes are generated such that the MAC is always >= 1.

Usage

```
GenGenoMat(n, snps, maf_range = c(0.001, 0.005))
```

Arguments

n	Sample size.
snps	Number of SNP in the gene.
maf_range	Range of minor allele frequencies: c(MIN, MAX).

Value

(n x snps) numeric matrix.

GenomicControl *Genomic Control*

Description

Genomic Control

Usage

```
GenomicControl(lambda, pval, df = 1)
```

Arguments

lambda	Genomic inflation factor.
pval	Numeric p-value.
df	Degrees of freedom. Should not require modification in most cases.

Value

Corrected p-value.

GenPheno *Generate Phenotypes*

Description

Simulate a phenotype based on annotations, covariates, and genotypes.

Usage

```
GenPheno(
  anno,
  beta,
  covar,
  geno,
  reg_param,
  binary = FALSE,
  include_residual = TRUE,
  indicator = FALSE,
  method = "none",
  prop_causal = 1,
  random_signs = FALSE,
  random_var = 0,
  weights = c(1, 1, 1)
)
```

Arguments

anno	(snps x 1) annotation vector.
beta	If method = "none", a (L x 1) coefficient with effect sizes for each annotation category. By default, there are L = 3 annotation categories corresponding to BMVs, DMVs, and PTVs. If method != "none", a scalar effect size for the allelic series burden score.
covar	Covariate matrix.
geno	(n x snps) genotype matrix.
reg_param	Regression parameters.
binary	Generate binary phenotype? Default: FALSE.
include_residual	Include residual? If FALSE, returns the expected value. Intended for testing.
indicator	Convert raw counts to indicators? Default: FALSE.
method	Genotype aggregation method. Default: "none".
prop_causal	Proportion of variants which are causal.
random_signs	Randomize signs? FALSE for burden-type genetic architecture, TRUE for SKAT-type.
random_var	Frailty variance in the case of random signs. Default: 0.
weights	Annotation category weights used for aggregation if method != "none".

Value

(n x 1) numeric vector.

Phenotype generation

- To generate phenotypes from the baseline model, set `method` to "none" and provide a vector `beta` of length equal to the number of annotation categories specifying the effect sizes of each.
- To generate phenotypes from the allelic series burden models, set `method` to "max" or "sum" and provide a scalar `beta`.
- To generate phenotypes from the allelic series SKAT model, set `method` to "none", set `random_signs` to true, and provide a vector `beta` of length equal to the number of annotation categories.

`isPD` *Check if Positive Definite*

Description

Check if Positive Definite

Usage

```
isPD(x, force_symmetry = FALSE, tau = 1e-08)
```

Arguments

x	Numeric matrix.
force_symmetry	Force the matrix to be symmetric?
tau	Threshold the minimum eigenvalue must exceed for the matrix to be considered positive definite.

Value

Logical indicating whether the matrix is PD.

OLS *Ordinary Least Squares*

Description

Fits the standard OLS model.

Usage

```
OLS(y, X)
```

Arguments

y	(n x 1) Numeric vector.
X	(n x p) Numeric matrix.

Value

List containing the following:

- beta: Regression coefficients.
- v: Residual variance.
- se: Standard errors.
- z: Z-scores.
- pval: P-values based on the chi2 distribution.

print.COAST *Print Method for COAST Object.*

Description

Print method for objects of class `COAST`.

Usage

```
## S3 method for class 'COAST'
print(x, ...)
```

Arguments

`x`	An object of class `COAST`.
`...`	Unused.

ResidVar *Calculate Residual Variance*

Description

Calculate Residual Variance

Usage

```
ResidVar(y, X)
```

Arguments

`y`	(n x 1) Numeric phenotype vector.
`X`	(n x q) Numeric covariate matrix.

Value

Scalar residual variance.

Score *Calculate Score Statistic*

Description

Calculate Score Statistic

Usage

```
Score(y, G, X, v)
```

Arguments

y	(n x 1) Numeric phenotype vector.
G	(n x p) Numeric genotype matrix.
X	(n x q) Numeric covariate matrix.
v	Scalar residual variance.

Value

Scalar score statistic.

show,COAST-method *Show Method for COAST Object*

Description

Show Method for COAST Object

Usage

```
## S4 method for signature 'COAST'
show(object)
```

Arguments

object	An object of class COAST.

`SumCountSS` *Allelic Sum Test from Sumstats*

Description

Allelic Sum Test from Sumstats

Usage

```
SumCountSS(anno, beta, ld, se, weights, return_beta = FALSE)
```

Arguments

`anno`	(snps x 1) annotation vector with integer values in 1 through the number of annotation categories L.
`beta`	(snps x 1) vector of effect sizes for the coding genetic variants within a gene.
`ld`	(snps x snps) matrix of correlations among the genetic variants.
`se`	(snps x 1) vector of standard errors for the effect sizes.
`weights`	(L x 1) vector of annotation category weights. Note that the number of annotation categories L is inferred from the length of `weights`.
`return_beta`	Return estimated effect sizes and standard errors? Default: FALSE.

Value

If `return_beta`, a list containing the category effect sizes, standard errors, and the p-value. Otherwise, the numeric p-value only.

Index