Package: AllelicSeries (via r-universe)

October 23, 2024

Title Allelic Series Test

Version 0.1.0.2

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

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>), Jianhui Gao [ctb] (<https://orcid.org/0000-0003-0915-1473>), insitro [cph]

Maintainer Zachary McCaw <zmccaw@insitro.com>

Repository CRAN

Date/Publication 2024-10-22 16:30:02 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 = DEFAULT_WEIGHTS
)
```

Arguments

`anno`	(snps x 1) annotation vector with values in c(0, 1, 2).
`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`	Annotation category weights.

Value

(n x 3) Numeric matrix without weighting, (n x 1) numeric matrix with weighting.

`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,
  score_test = FALSE,
  weights = DEFAULT_WEIGHTS
)
```

Arguments

`anno`	(snps x 1) annotation vector with values in c(0, 1, 2).
`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.
`score_test`	Run a score test? If FALSE, performs a Wald test.
`weights`	(3 x 1) annotation category weights.

Value

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,
  maf = NULL,
  method = "none",
  weights = DEFAULT_WEIGHTS
)
```

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.
`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.
`method`	Method for aggregating across categories: ("none", "sum"). Default: "none".
`weights`	(3 x 1) vector of annotation category weights.

Value

Numeric p-value of the allelic series burden test.

Examples

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

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

Arguments

`anno`	(snps x 1) annotation vector with values in c(0, 1, 2).
`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`	(3 x 1) annotation category 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 = DEFAULT_WEIGHTS
)
```

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.
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	(3 x 1) vector of annotation category weights.

Value

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

Examples

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

# Run allelic series SKAT test from sumstats.
results <- ASKATSS(
  anno = sumstats$anno,
  beta = sumstats$sumstats$beta,
  maf = sumstats$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)
```

Arguments

`anno`	(snps x 1) annotation vector.
`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.

Value

Numeric p-value.

`CalcRegParam` *Calculate Regression Parameters*

Description

Calculate phenotypic regression coefficients and the residual variation based on proportion of variation explained (PVE) by each factor. Note that the proportion of variation explained by genotype is required, but genetic effects are not generated here.

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

Calculate Summary Statistics

Usage

```
CalcSumstats(
  anno = NULL,
  covar = NULL,
  data = NULL,
  geno = NULL,
  pheno = NULL,
  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.
`is_binary`	Is the phenotype binary? Default: FALSE.

Value

List containing the following items:

- anno: A SNP-length annotation vector.
- ld: A SNP x SNP correlation (LD) matrix.
- maf: Minor allele frequency of each variant.
- sumstats: A SNP x 4 matrix of summary statistics.
- 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`	(3 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, maf)
```

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.
`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,
  apply_int = TRUE,
  covar = NULL,
  include_orig_skato_all = FALSE,
  include_orig_skato_ptv = FALSE,
  is_pheno_binary = FALSE,
  min_mac = 0,
  pval_weights = NULL,
  return_omni_only = FALSE,
  score_test = FALSE,
  weights = DEFAULT_WEIGHTS
)
```

Arguments

`anno`	(snps x 1) annotation vector with values in c(0, 1, 2).
`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.
`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.
`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`	(3 x 1) annotation category weights.

Value

Numeric p-value.

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

`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),
  maf = NULL,
  ld = NULL,
  pval_weights = c(1, 1, 2),
  weights = DEFAULT_WEIGHTS
)
```

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.
`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.
`maf`	(snps x 1) vector of minor allele frequencies. Although ideally provided, defaults to the zero vector.
`ld`	(snps x snps) matrix of correlations among the genetic variants. Although ideally provided, an identity matrix is assumed if not.
`pval_weights`	(3 x 1) vector of relative weights for combining the component tests to perform the omnibus test.
`weights`	(3 x 1) vector of annotation category weights. The default of c(1, 1, 2) gives the SKAT test equal weight to the two burden tests.

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$anno,
  beta = sumstats$sumstats$beta,
  maf = sumstats$maf,
  se = sumstats$sumstats$se,
  ld = sumstats$ld
)
show(results)
```

`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
)
```

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, min_mac = 0L)
```

Arguments

`anno`	(snps x 1) annotation vector with values in c(0, 1, 2).
`geno`	(n x snps) genotype matrix.
`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(0, 1, 2),
  binary = FALSE,
  geno = NULL,
  include_residual = TRUE,
  indicator = FALSE,
  maf_range = c(0.005, 0.01),
  method = "none",
  n = 100,
  p_dmv = 0.4,
  p_ptv = 0.1,
  prop_causal = 1,
  random_signs = FALSE,
  random_var = 0,
  snps = 100,
  weights = c(1, 2, 3)
)
```

Arguments

`anno`	Annotation vector, if providing genotypes. Should match the number of columns in geno.
`beta`	If method = "none", a (3 x 1) coefficient vector for bmvs, dmvs, and ptvs respectively. If method != "none", a scalar effect size.
`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.
`p_dmv`	Frequency of deleterious missense variants. Default of 40% is based on the frequency of DMVs among rare coding variants in the UK Biobank.
`p_ptv`	Frequency of protein truncating variants. Default of 10% is based on the frequency of PTVs among rare coding variants in the UK Biobank.
`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`	Aggregation weights.

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)
```

`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 classified as a benign missense variant (0), a deleterious missense variant (1), or a protein truncating variant (2).

Usage

```
GenAnno(snps, p_dmv = 0.33, p_ptv = 0.33)
```

Arguments

`snps`	Number of SNPs in the gene.
`p_dmv`	Frequency of deleterious missense variants.
`p_ptv`	Frequency of protein truncating variants.

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

Generate Genotypes

Usage

```
GenGeno(n, snps, maf_range = c(0.005, 0.01), p_dmv = 0.33, p_ptv = 0.33)
```

Arguments

`n`	Sample size.
`snps`	Number of SNP in the gene.
`maf_range`	Range of minor allele frequencies: c(MIN, MAX).
`p_dmv`	Frequency of deleterious missense variants.
`p_ptv`	Frequency of protein truncating variants.

Value

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

`GenGenoMat` *Generate Genotype Matrix*

Description

Generate Genotype Matrix

Usage

```
GenGenoMat(n, snps, maf_range = c(0.005, 0.01))
```

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

Generate Phenotypes

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(0, 1, 2)
)
```

Arguments

`anno`	(snps x 1) annotation vector.
`beta`	(3 x 1) coefficient vector for bmvs, dmvs, and ptvs respectively.
`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`	Aggregation weights.

Value

(n x 1) numeric vector.

---

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)
```

Arguments

`anno`	(snps x 1) annotation vector.
`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`	(3 x 1) vector of annotation category weights.

Value

Numeric p-value.

Index