Package 'gnonadd'

September 22, 2023

Type Package

Title Various Non-Additive Models for Genetic Associations

Version 1.0.2

Description The goal of 'gnonadd' is to simplify workflows in the analysis of non-additive effects of sequence variants. This includes variance effects (Ivarsdottir et. al (2017) <doi:10.1038/ng.3928>), correlation effects, interaction effects and dominance effects. The package also includes convenience functions for visualization.

URL `https://github.com/DecodeGenetics/gnonadd`

BugReports `https://github.com/DecodeGenetics/gnonadd/issues`

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.1

Depends R (>= 2.10)

Imports ggplot2

Suggests MASS

NeedsCompilation no

Author Audunn S. Snaebjarnarson [aut, cre, ctb],
Gudmundur Einarsson [aut, ctb],
Daniel F. Gudbjartsson [aut, ctb],
deCODE Genetics/AMGEN [cph, fnd]

Maintainer Audunn S. Snaebjarnarson <`audunn.snaebjarnarson@decode.is`>

Repository CRAN

Date/Publication 2023-09-22 11:30:02 UTC

R topics documented:

`alpha.calc` *Genetic variance effects*

Description

This function estimates the variance effect of a genetic variant on a quantitative trait. The genotype is coded as 0 (non-carrier), 1 (single copy of effect allele) and 2 (two copies of effect allele). The variance effect (alpha) is modelled to be multiplicitive. We use a likelihood ratio test with 1 degree of freedom, * H0: y~N(mu_g,sigma^2) * H1: y~N(mu_g,sigma^2*alpha^g) Under the alternative model, the variance of the trait changes multiplicatively with genotype: Var(y|g=j)=alpha^j*sigma^2, j in 0,1,2

Usage

```
alpha.calc(qt, g)
```

Arguments

`qt`	A numeric vector.
`g`	An integer vector.

Value

A list with the values:

* alpha, the estimated variance effect * sigma2_alt, The variance estimated for non-carriers * pval, the p-value of the likelihood ratio test * chi2, the chi squared statistics (with one degree of freedom) corrisponding to the p-value

Examples

```
n_val <- 50000L
geno_vec <- sample(c(0, 1, 2), size = n_val, replace = TRUE)
qt_vec <- rnorm(n_val) * (1.1^geno_vec)
res <- alpha.calc(qt_vec, geno_vec)
```

`alpha.cond` *Conditional analysis for genetic variance effects*

Description

We estimate the variance effect of a primary variant conditioned on one or more secondary variants. We apply a likelyhood ratio test with one degree of freedom

H0: All secondary variants have a variance effect, but not the primary one. H1: All variants have a variance effect, including the primary one

Under both models we assume a full mean effect model. That is, the number of mean-value parameters is 3^(n+1), where n is the number of covariates Thus the null model has 3^(n+1)+n degrees of freedom whereas the alternative model has 3^(n+1)+n+1 Due to the exponential growth of free parameters, the test might have low statistical power if many covariates are used

IMPORTANT NOTE: We use the Gauss-Newton algorithm to estimate many parameters. We do not check if the algorithm converges to the true minimum values, or if it converges at all. Thus, we advise against blindly believing all results

Usage

```
alpha.cond(qt, g, g_covar, iter_num = 50)
```

Arguments

`qt`	A numeric vector.
`g`	An integer vector.
`g_covar`	An integer matrix where each column corresponds to a genetic covariate
`iter_num`	An integer. Represents the number of iterations performed in the Gauss-Newton algorithm

Value

A list with the values: * alpha, the estimated variance effect, conditioned on the covariates * pval, the p-value corresponding to alpha

Examples

```
n <- 10000
qt <- rnorm(n)
g <- rbinom(n, 2, 0.3)
qt <- qt * 1.2^g
g_covar <- as.data.frame(matrix(0, nrow = n, ncol = 3))
for(i in 1:ncol(g_covar)){
  freq <- runif(1, min = 0, max = 1)
  g_covar[, i] <- rbinom(n, 2, freq)
}
res <- alpha.cond(qt, g, g_covar)
```

`alpha.continuous.cond` *variance effect conditioned on continuous variables*

Description

We estimate the variance effect of a variant conditioned on one or more continous variable We apply a likelyhood ratio test with one degree of freedom

H0: All covariates have a variance effect, but not the variant H1: The variant has a variance effect, and the covariates as well

Usage

```
alpha.continuous.cond(qt, g, x, iter_num = 50, eps_param = 1e-10)
```

Arguments

`qt`	A numeric vector.
`g`	An integer vector.
`x`	A numeric matrix, each column represents a covariate.
`iter_num`	An integer. Represents the number of iterations performed in the Gauss-Newton algorithm
`eps_param`	A number. The Gauss-Newton algorithm terminates if the incriment change of all variance estimates is smaller than this number.

Value

A list with the values: * alpha, the estimated variance effect, conditioned on the covariates * pval, the p-value corresponding to alpha

Examples

```
n_val <- 50000
x <- matrix(0,nrow = n_val, ncol = 4)
for(i in 1:4) {
x[, i] <- rnorm(n_val)
}
g_vec <- rbinom(n_val,2,0.3)
var_vec <- exp(0.2 * x[, 1] - 0.3 * x[, 4] + 0.3 * g_vec)
qt_vec <- rnorm(n_val, 0, sqrt(var_vec))
res <- alpha.continuous.cond(qt_vec, g_vec, x)
```

`alpha.multi.est` *Variance parameters*

Description

This function jointly estimates the variance effect of a set of (continuous) variables on a qt trait. More precisely. It finds the maximum likelyhood estimators.

Usage

```
alpha.multi.est(
  qt,
  x,
  iter_num = 50,
  eps_param = 1e-10,
  initial_guess = rep(0, ncol(x))
)
```

Arguments

`qt`	A numeric vector.
`x`	A numeric matrix, each column represents a covariate.
`iter_num`	An integer. Represents the number of iterations performed in the Gauss-Newton algorithm
`eps_param`	A number. The Gauss-Newton algorithm terminates if the incriment change of all variance estimates is smaller than this number.
`initial_guess`	A vector of length ncol(x). Represents the initial guess of parameters for the Gauss-Newton algorithm.

Value

A vector with a variance estimate for each variable.

Examples

```
n_val <- 50000
x <- matrix(0,nrow = n_val, ncol = 4)
for(i in 1:4) {
x[, i] <- rnorm(n_val)
}
var_vec <- exp(0.2 * x[, 1] - 0.3 * x[, 4])
qt_vec <- rnorm(n_val, 0, sqrt(var_vec))
res <- alpha.multi.est(qt_vec, x)
```

---

corr.calibration *Calibration for the correlation test*

Description

This function finds appropriate scaling_parameters for the kappa_calc function by means of bootstrapping.

Usage

```
corr.calibration(qt1, qt2, reps = 10000)
```

Arguments

qt1	A numeric vector.
qt2	A numeric vector.
reps	The number of repeates we want to perform for each sample size

Value

A list with the values:

* bias_scale, a value to determine the bias for the correlation estimators * weight_scale, a value to determine how much weight we assign to each correlation estimator * safe_weight_scale, same as weight scale, but larger. Using this weight decreases the chance of type 1 error, with the cost of statistical power.

Examples

```
n_val <- 10000
Q <- MASS::mvrnorm(n = n_val, mu = c(0,0), Sigma = matrix(c(1,0.3,0.3,1),
                   nrow = 2, ncol = 2))
qt1_val <- Q[,1]
qt2_val <- Q[,2]
res <- corr.calibration(qt1_val, qt2_val, 10)
```

`dominance.calc` *Genetic dominance effects*

Description

This function estimates the dominance effect of a genetic variant on a quantitatvie trait Nothing fancy here. We apply a simple linear regression model to estimate dominance effects. We include a linear term, coded as 0,1 and 2 for non-carriers, heterozygotes and homozygous carriers of the effect allele. We also include a dominance term, coded as 1 for homozygous carriers and 0 for others. Effect size and significance is based on the dominance term.

Usage

```
dominance.calc(
  qt,
  g,
  round_imputed = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`qt`	A numeric vector
`g`	A vector with (possibly imputed) genotype values. All entries should be larger than 0 and smaller than 2.
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis
`covariates`	A dataframe containing any covariates that should be used; one column per covariate.

Value

A list with the dominanc effect and corresponding standard error, t statistic and p-value

Examples

```
g_vec <- rbinom(100000, 2, 0.3)
qt_vec <- rnorm(100000) + 0.2 * g_vec^2
res <- dominance.calc(qt_vec, g_vec)
```

dominance_CC.calc *Genetic dominance effects on a case control variable*

Description

This function estimates the dominance effect of a genetic variant on a case-control variable We apply a logistic regression model to estimate dominance effects. We include a linear term, coded as 0,1 and 2 for non-carriers, heterozygotes and homozygous carriers of the effect allele. We also include a dominance term, coded as 1 for homozygous carriers and 0 for others. Effect size and significance is based on the dominance term.

Usage

```
dominance_CC.calc(
  cc,
  g,
  yob = rep(-1, length(cc)),
  sex = rep(-1, length(cc)),
  round_imputed = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`cc`	A case control vector, containing 0's and 1's
`g`	A vector with (possibly imputed) genotype values. All entries should be larger than 0 and smaller than 2.
`yob`	A numerical vector containing year of birth. If some are unknown they should be marked as -1
`sex`	A numerical vector containing sex, coded 0 for males, 1 for females and -1 for unknown
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis
`covariates`	A dataframe containing any other covariates that should be used; one column per covariate.

Value

A list with the dominanc effect (on log-scale) and corresponding standard error, z statistic and p-value

Examples

```
g_vec <- rbinom(100000, 2, 0.3)
cc_vec <- rbinom(100000, 1, 0.1 * (1.2 ^ (g_vec^2)))
res <- dominance_CC.calc(cc_vec, g_vec)
```

`ellipse.by.gen` *Ellipse best fit plot*

Description

This tool creates a scatter plot along with regression lines. Additionally it finds and plots the best ellipses that fit the data.

Usage

```
ellipse.by.gen(
  qt1,
  qt2,
  g,
  trait_name1 = "qt trait 1",
  trait_name2 = "qt trait 2",
  title = "",
  sample_size = 500
)
```

Arguments

`qt1`	A numeric vector.
`qt2`	A numeric vector.
`g`	An integer vector.
`trait_name1`	A string.
`trait_name2`	A string.
`title`	A string.
`sample_size`	A positive integer.

Value

A scatter plot.

Examples

```
n_val <- 10000L
geno_vec <- c(rep(0, n_val), rep(1, n_val), rep(2, n_val))
qt_g0 <- MASS::mvrnorm(n_val, mu = c(0, 0), Sigma = matrix(c(0.93, 0.88, 0.88, 0.92), ncol = 2))
qt_g1 <- MASS::mvrnorm(n_val, mu = c(0, 0), Sigma = matrix(c(0.98, 0.88, 0.88, 0.90), ncol = 2))
qt_g2 <- MASS::mvrnorm(n_val, mu = c(0, 0), Sigma = matrix(c(1.57, 0.81, 0.81, 0.59), ncol = 2))
qt_vec <- rbind(qt_g0, qt_g1)
qt_vec <- rbind(qt_vec, qt_g2)
res <- ellipse.by.gen(qt_vec[, 1], qt_vec[, 2], geno_vec)
```

`env_interaction.calc` *Variant-Environmental interaction effects*

Description

This function estimates the interaction effect of a genetic variant with an environmental factor on a quantitatvie trait We apply a simple linear regression model to estimate interaction effects. We include a linear term for the variant and environmental variable seperately. The variant is coded as 0,1 and 2 for non-carriers, heterozygotes and homozygous carriers of the effect allele. The environmental variable is rank-normalized automatically as part of the function. The interaction term is defined as the product of the genetic and the (normalized) environmental variables. Effect size and significance is based on the interaction term.

Usage

```
env_interaction.calc(
  qt,
  g,
  env,
  round_imputed = FALSE,
  dominance_term = FALSE,
  square_env = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`qt`	A numeric vector
`g`	A vector with (possibly imputed) genotype values. All entries should be larger than 0 and smaller than 2.
`env`	A numeric vector with an environmental variable
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
`dominance_term`	A boolian variable determining whether a dominance term for the variant should be included as a covariates in the analysis
`square_env`	A boolian variable determining whether the square of the environmental trait should be included as a covariate in the analysis
`covariates`	A dataframe containing any other covariates that should be used; one column per covariate

Value

A list with the environmental interaction effect and corresponding standard error, t statistic and p-value

Examples

```
g_vec <- rbinom(100000, 2, 0.1)
env_vec <- round(runif(100000,min=0,max=6))
qt_vec <- rnorm(100000) + 0.1 * g_vec + 0.05 * env_vec + 0.05 * g_vec * env_vec
res <- env_interaction.calc(qt_vec, g_vec, env_vec)
```

`env_interaction_CC.calc` *Variant-Environmental interaction effects on a case control variable*

Description

This function estimates the interaction effect of a genetic variant with an environmental factor on a case control variable We apply a simple logistic regression model to estimate interaction effects. We include a linear term for the variant and environmental variable seperately. The variant is coded as 0,1 and 2 for non-carriers, heterozygotes and homozygous carriers of the effect allele. The environmental variable is rank-normalized automatically as part of the function. The interaction term is defined as the product of the genetic and the (normalized) environmental variable. Effect size and significance is based on the interaction term.

Usage

```
env_interaction_CC.calc(
  cc,
  g,
  env,
  yob = rep(-1, length(cc)),
  sex = rep(-1, length(cc)),
  round_imputed = FALSE,
  dominance_term = FALSE,
  square_env = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`cc`	A numeric vector
`g`	A vector with (possibly imputed) genotype values. All entries should be larger than 0 and smaller than 2.
`env`	A numeric vector with an environmental variable
`yob`	A numerical vector containing year of birth. If some are unknown they should be marked as -1
`sex`	A numerical vector containing sex, coded 0 for males, 1 for females and -1 for unknown
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.

`dominance_term`	A boolian variable determining whether a dominance term for the variant should be included as a covariates in the analysis
`square_env`	A boolian variable determining whether the square of the environmental trait should be included as a covariate in the analysis
`covariates`	A dataframe containing any other covariates that should be used; one column per covariate

Value

A list with the environmental interaction effect and corresponding standard error, t statistic and p-value

Examples

```
g_vec <- rbinom(100000, 2, 0.9)
env_vec <- round(runif(100000, min = 0, max = 6))
cc_vec <- rbinom(100000,1,0.1 * (1.05^g_vec) *
          (1.1^env_vec)* (1.1 ^ (g_vec * env_vec)))
res <- env_interaction_CC.calc(cc_vec, g_vec, env_vec)
```

`expected.variance.effect` *Expected variance effect from additive effect*

Description

This function interpolates data from a simple simulation to give an estimate of the variance effect induced by an additive effect. The simulation code is stored under inst/raw/. We assume that the trait has been inverse normal transformed. Under the simulation, there is no variance effect, so the variance effect is fully induced by the inverse normal transform.

Usage

```
expected.variance.effect(maf, beta_add)
```

Arguments

`maf`	Minor allele frequency of the variant, should be in the range 0 to 0.5.
`beta_add`	Additive effect of the variant, should be in the range 0 to 3.5. This variable can be a vector of values.

Value

The expected variance effect for the variant from the given maf, beta combination.

Examples

```
maf <- 0.1
beta_val <- 0.3
expected_var <- expected.variance.effect(maf, beta_val)

beta_vec <- seq(0.1,2, length.out = 20)
expected_var <- expected.variance.effect(maf, beta_vec)
```

`hist_by_gen` *Histogram by genotype*

Description

This tool creates three histogram plots. One per genotype. Additionally, outliers are colored red (by default subjects that are in the top and bottom 2.5 and blue lines are added to indicate the mean and (by default) one standard deviation in each direction.

Usage

```
hist_by_gen(
  qt,
  g,
  bins = 100,
  trait_name = "qt trait",
  title = "",
  outlier_quantiles = c(0.025, 0.975),
  sd_lines = c(1, 1)
)
```

Arguments

`qt`	A numeric vector.
`g`	An integer vector.
`bins`	An integer.
`trait_name`	A string.
`title`	A string.
`outlier_quantiles`	A vector with length 2.
`sd_lines`	A vector with length 2.

Value

A histgram plot

Examples

```
n_val <- 50000L
geno_vec <- sample(c(0, 1, 2), size = n_val, replace = TRUE)
qt_vec <- rnorm(n_val) * (1.3^geno_vec) + 1 * geno_vec
hist_by_gen(qt_vec, geno_vec)
```

`interaction.calc` *Variant-Variant interaction effects*

Description

This function estimates the interaction effect of a pair of genetic variant on a quantitatvie trait We apply a simple linear regression model to estimate interaction effects. We include a linear term for each variant seperately, coded as 0,1 and 2 for non-carriers, heterozygotes and homozygous carriers of the effect allele. We also include an interaction term, coded as the product of the two genotype values. Effect size and significance is based on the interaction term.

Usage

```
interaction.calc(
  qt,
  g1,
  g2,
  round_imputed = FALSE,
  dominance_terms = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`qt`	A numeric vector
`g1`	A vector with (possibly imputed) genotype values. All entries should be larger than or equal to 0 and smaller than or equal to 2.
`g2`	A vector with (possibly imputed) genotype values. All entries should be larger than or equal to 0 and smaller than or equal to 2.
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
`dominance_terms`	A boolian variable determining whether dominance terms for the variants should be included as covariates in the analysis
`covariates`	A dataframe containing any other covariates that should be used; one column per covariate

Value

A list with the interaction effect and corresponding standard error, t statistic and p-value

Examples

```
g1_vec <- rbinom(100000, 2, 0.9)
g2_vec <- rbinom(100000, 2, 0.1)
qt_vec <- rnorm(100000) + 0.1 * g1_vec + 0.2 * g2_vec +0.4 * g1_vec * g2_vec
res <- interaction.calc(qt_vec, g1_vec, g2_vec)
```

`interaction_CC.calc` *Variant-Variant interaction effects on a case control variable*

Description

This function estimates the interaction effect of a pair of genetic variant on a case-control variable We apply a logistic regression model to estimate interaction effects. We include a linear term for each variant seperately, coded as 0,1 and 2 for non-carriers, heterozygotes and homozygous carriers of the effect allele. We also include an interaction term, coded as the product of the two genotype values. Effect size and significance is based on the interaction term.

Usage

```
interaction_CC.calc(
  cc,
  g1,
  g2,
  yob = rep(-1, length(cc)),
  sex = rep(-1, length(cc)),
  round_imputed = FALSE,
  dominance_terms = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`cc`	A numeric vector
`g1`	A vector with (possibly imputed) genotype values. All entries should be larger than 0 and smaller than 2.
`g2`	A vector with (possibly imputed) genotype values. All entries should be larger than 0 and smaller than 2.
`yob`	A numerical vector containing year of birth. If some are unknown they should be marked as -1
`sex`	A numerical vector containing sex, coded 0 for males, 1 for females and -1 for unknown
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
`dominance_terms`	A boolian variable determining whether dominance terms for the variants should be included as covariates in the analysis

`covariates`	A dataframe containing any other covariates that should be used; one column per covariate

Value

A list with the interaction effect (on log-scale) and corresponding standard error, z statistic and p-value

Examples

```
g1_vec <- rbinom(100000, 2, 0.9)
g2_vec <- rbinom(100000, 2, 0.1)
cc_vec <- rbinom(100000,1,0.1 * (1.05^g1_vec) *
          (1.05^g2_vec) * (1.3 ^ (g1_vec * g2_vec)))
res <- interaction_CC.calc(cc_vec, g1_vec, g2_vec)
```

`kappa_calc` *Genetic correlation effects*

Description

This function estimates the correlation effect of a genetic variant on a pair of quantitative traits. The effect (kappa) is measured on Fisher transformed correlation values. The genotype is coded as 0 (non-carrier), 1 (single copy of effect allele) and 2 (two copies of effect allele). We use a likelihood ratio test with 1 degree of freedom: * H0: z_0 = z_1 = z_2, * H1: z_j = z_null +kappa*j, where z_j is the Fisher-transformed correlation between the traits amongst subjects of genotype j.

If the traits follow a joint normal distribution, then (thoretically) the Fisher-transformed sample correlation (z) is approximately normally distributed with mean 0 and standard error 1/sqrt(n-3). Otherwise, we have to correct for possible bias in the estimators and scale the weights we assign to them appropriately

Note, that even though each trait follows a normal distribution individually, that does not necessarily imply that the pair of them follow a joint normal distirbution.

Usage

```
kappa_calc(qt1, qt2, g, weight_scale = 1, bias_scale = 0)
```

Arguments

`qt1`	A numeric vector.
`qt2`	A numeric vector.
`g`	An integer vector.
`weight_scale`	Used to appropriately scale the weight assigned to the correlation estimators
`bias_scale`	Used to appropriately scale the bias of the correlation estimators

Value

A list with the values:

* kappa, the estimated correlation effect * pval, the p-value of the likelihood ratio test

Examples

```
Sigma0 <- matrix(c(1,0,0,1),nrow=2,ncol=2)
Sigma1 <- matrix(c(1,0.3,0.3,1),nrow=2,ncol=2)
Sigma2 <- matrix(c(1,0.6,0.6,1),nrow=2,ncol=2)
geno_vec <- c(rep(0,10000),rep(1,1000),rep(2,100))
Q0 <- MASS::mvrnorm(n = 10000, mu = c(0,0), Sigma = Sigma0)
Q1 <- MASS::mvrnorm(n = 1000, mu = c(0,0), Sigma = Sigma1)
Q2 <- MASS::mvrnorm(n = 100, mu = c(0,0), Sigma = Sigma2)
Q <- rbind(Q0,Q1)
Q <- rbind(Q,Q2)
qt1_val <- Q[,1]
qt2_val <- Q[,2]
res <- kappa_calc(qt1_val, qt2_val, geno_vec)
```

---

pairwise_env_int.calc *Pairwise environmental interaction effects*

Description

Given a set of variants and environmental traits, and a single quantitative trait, this function calculates the interaction effect of all possible variant-environmental pairs

Usage

```
pairwise_env_int.calc(
  qt,
  g,
  env,
  round_imputed = FALSE,
  dominance_term = FALSE,
  square_env = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0)),
 variant_names = paste(rep("variant", ncol(g)), as.character(1:ncol(g)), sep = "_"),
  env_names = paste(rep("env", ncol(env)), as.character(1:ncol(env)), sep = "_")
)
```

Arguments

qt	A numeric vector
g	A matrix, where each colomn represents a variant
env	A matrix, where each row represents an environmental variable

`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
`dominance_term`	A boolian variable determining whether a dominance term for the variant should be included as a covariates in the analysis
`square_env`	A boolian variable determining whether the square of the environmental trait should be included as a covariate in the analysis
`covariates`	A dataframe containing any other covariates that should be used; one column per covariate
`variant_names`	A list of the names of the variants
`env_names`	A list of the names of the environmental variables

Value

A dataframe with all possible variant-environmental pairs and their estimated interaction effect

Examples

```
g_vec <- matrix(0, nrow = 100000, ncol = 3)
freqs <- runif(ncol(g_vec), min = 0, max = 1)
env_vec <- matrix(0, nrow = 100000, ncol = 3)
for(i in 1:ncol(g_vec)){
 g_vec[, i] <- rbinom(100000, 2, freqs[i])
}
for( i in 1:ncol(env_vec)){
 env_vec[, i] <- round(runif(100000,min=0,max=6))
}

qt_vec <- rnorm(100000) + 0.1 * g_vec[, 1] + 0.2 *
          g_vec[, 2] -0.1 * env_vec[, 3] + 0.1 *
          env_vec[, 1] + 0.1 * g_vec[, 1] * env_vec[, 1]
res <- pairwise_env_int.calc(qt_vec, g_vec, env_vec)
```

---

pairwise_env_int_CC.calc

Pairwise environmental interaction effects for a case control variable

---

Description

Given a set of variants and environmental traits, and a single case control variable, this function calculates the interaction effect of all possible variant-environmental pairs

Usage

```
pairwise_env_int_CC.calc(
  cc,
  g,
  env,
  yob = rep(-1, length(cc)),
  sex = rep(-1, length(cc)),
  round_imputed = FALSE,
  dominance_term = FALSE,
  square_env = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0)),
  variant_names = paste(rep("variant", ncol(g)), as.character(1:ncol(g)), sep = "_"),
  env_names = paste(rep("env", ncol(env)), as.character(1:ncol(env)), sep = "_")
)
```

Arguments

cc	A numeric vector
g	A matrix, where each colomn represents a variant
env	A matrix, where each row represents an environmental variable
yob	A numerical vector containing year of birth. If some are unknown they should be marked as -1
sex	A numerical vector containing sex, coded 0 for males, 1 for females and -1 for unknown
round_imputed	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
dominance_term	A boolian variable determining whether a dominance term for the variant should be included as a covariates in the analysis
square_env	A boolian variable determining whether the square of the environmental trait should be included as a covariate in the analysis
covariates	A dataframe containing any other covariates that should be used; one column per covariate
variant_names	A list of the names of the variants
env_names	A list of the names of the environmental variables

Value

A dataframe with all possible variant-environmental pairs and their estimated interaction effect

Examples

```
N_run <- 25000
g_vec <- matrix(0, nrow = N_run, ncol = 3)
freqs <- runif(ncol(g_vec), min = 0, max = 1)
env_vec <- matrix(0, nrow = N_run, ncol = 3)
for(i in 1:ncol(g_vec)){
 g_vec[, i] <- rbinom(N_run, 2, freqs[i])
}
for( i in 1:ncol(env_vec)){
 env_vec[, i] <- round(runif(N_run,min=0,max=6))
}
cc_vec <- rbinom(N_run,1,0.1 * (1.05 ^ g_vec[, 1]) *
          (1.06 ^ env_vec[,1]) * (0.95 ^ g_vec[, 2]) *
          (1.1^(g_vec[, 1] * env_vec[, 1])))
res <- pairwise_env_int_CC.calc(cc_vec, g_vec, env_vec)
```

`pairwise_int.calc` *Pairwise interaction effects*

Description

Given a set of variants and a quantitative trait, this function calculates the interaction effect of all possible variant-variant pairs

Usage

```
pairwise_int.calc(
  qt,
  g,
  round_imputed = FALSE,
  dominance_terms = FALSE,
 variant_names = paste(rep("variant", ncol(g)), as.character(1:ncol(g)), sep = "_"),
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0))
)
```

Arguments

`qt`	A numeric vector
`g`	A matrix, where each colomn represents a variant
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
`dominance_terms`	A boolian variable determining whether dominance terms for the variants should be included as covariates in the analysis
`variant_names`	A list of the names of the variants
`covariates`	A dataframe containing any other covariates that should be used; one column per covariate

Value

A dataframe with all possible variant pairs and their estimated interaction effect

Examples

```
g_vec <- matrix(0, nrow = 100000, ncol = 5)
freqs <- runif(ncol(g_vec), min = 0, max = 1)
for(i in 1:ncol(g_vec)){
 g_vec[,i] <- rbinom(100000, 2, freqs[i])
}

qt_vec <- rnorm(100000) + 0.1 * g_vec[, 1] + 0.2 *
          g_vec[, 2] -0.1 * g_vec[, 3] + 0.2 *
          g_vec[, 1] * g_vec[, 2]
res <- pairwise_int.calc(qt_vec, g_vec)
```

`pairwise_int_CC.calc` *Pairwise interaction effects for a case control variable*

Description

Given a set of variants and a case control variable, this function calculates the interaction effect of all possible variant-variant pairs

Usage

```
pairwise_int_CC.calc(
  cc,
  g,
  yob = rep(-1, length(cc)),
  sex = rep(-1, length(cc)),
  round_imputed = FALSE,
  dominance_terms = FALSE,
  covariates = as.data.frame(matrix(0, nrow = 0, ncol = 0)),
 variant_names = paste(rep("variant", ncol(g)), as.character(1:ncol(g)), sep = "_")
)
```

Arguments

`cc`	A numeric vector
`g`	A matrix, where each colomn represents a variant
`yob`	A numerical vector containing year of birth. If some are unknown they should be marked as -1
`sex`	A numerical vector containing sex, coded 0 for males, 1 for females and -1 for unknown
`round_imputed`	A boolian variable determining whether imputed genotype values should be rounded to the nearest integer in the analysis.
`dominance_terms`	A boolian variable determining whether dominance terms for the variants should be included as covariates in the analysis

`covariates`	A dataframe containing any other covariates that should be used; one column per covariate
`variant_names`	A list of the names of the variants

Value

A dataframe with all possible variant pairs and their estimated interaction effect

Examples

```
N_run <- 25000
g_vec <- matrix(0, nrow = N_run, ncol = 5)
freqs <- runif(ncol(g_vec), min = 0,max = 1)
for(i in 1:ncol(g_vec)){
 g_vec[, i] <- rbinom(N_run, 2, freqs[i])
}
cc_vec <- rbinom(N_run,1,0.1 * (1.05 ^ g_vec[, 1]) *
          (1.06 ^ g_vec[,2]) * (0.95 ^ g_vec[, 3]) *
          (1.5^(g_vec[,1] * g_vec[,2])))
res <- pairwise_int_CC.calc(cc_vec, g_vec)
```

`PRS_creator` *Creates poligenic risk scores*

Description

This function creates genetic risk scores, with the option of including interactions or dominance effects. The score is automatically shifted to have mean 0.

Usage

```
PRS_creator(
  g,
  betas,
  dominance_effects = rep(0, length(betas)),
  interaction_effects = matrix(0, nrow = 0, ncol = 0),
  log_scale = FALSE
)
```

Arguments

`g`	A matrix, where each colomn represents a variant and each line represents a subject
`betas`	A numeric vector, representing the (additive) effects of the variants
`dominance_effects`	A numeric vector, representing dominance effects of the variants

`interaction_effects`	A matrix with three columns. First two columns are integers that correspond to the variants that are interacting. The third column is the effect size.
`log_scale`	A Boolian variabe. If true all analysis is done on log-transformed effect values and the resulting score is transformed back to an exponential scale in the end.

Value

A numeric vector with a poligenic risk score for each subject

Examples

```
g_vec <- matrix(0, nrow = 100000, ncol = 5)
freqs <- runif(ncol(g_vec), min = 0, max = 1)
for(i in 1:ncol(g_vec)){
 g_vec[,i] <- rbinom(100000, 2, freqs[i])
}
beta_vec <- runif(5, min = -0.5, max = 0.5)
dom_vec <- runif(5, min = -0.5, max = 0.5)
int_vec <- matrix(0,nrow = 2, ncol = 3)
int_vec[, 1] <- c(1, 3)
int_vec[, 2] <- c(2, 5)
int_vec[, 3] <- runif(2, min = -0.5, max = 0.5)
res <- PRS_creator(g_vec, beta_vec, dominance_effects = dom_vec, interaction_effects = int_vec)
```

`train_and_impute_PRS` *Trains and imputes a poligenic risk score (PRS)*

Description

This function trains a poligenic risk score model on a dataset, and then imputes the risk score into another dataset

Usage

```
train_and_impute_PRS(
  qt_training,
  g_training,
  g_impute,
  dominance_effects = rep(FALSE, ncol(g_training)),
  interaction_effects = matrix(0, nrow = 0, ncol = 0)
)
```

Arguments

`qt_training`	A numeric vector. Represents the qt values of the data we train the model on.
`g_training`	A matrix, where each colomn represents a variant and each line represents a subject in the training data

`g_impute`	A matrix, where each column represents a variant and each line represents a subject.
`dominance_effects`	A Boolian vector, each term determines whether a dominance term for the corresponding variant is used in the model.
`interaction_effects`	An integer matrix with two columns. Each line represents a pair of interacting variants that should be included in the model.

Value

Returns a list with the following objects * PRS_imputed, the imputed PRS values * PRS_training, the PRS values for the training data * Residuals_training, the residuals from the model in the training data

Examples

```
g_train_vec <- matrix(0, nrow = 100000, ncol = 5)
freqs <- runif(ncol(g_train_vec), min = 0, max = 1)
for(i in 1:ncol(g_train_vec)){
 g_train_vec[,i] <- rbinom(100000, 2, freqs[i])
}
g_impute_vec <- matrix(0, nrow = 50000, ncol = 5)
for(i in 1:ncol(g_impute_vec)){
 g_impute_vec[,i] <- rbinom(50000, 2, freqs[i])
}
dom_vec <- c(TRUE, FALSE, FALSE, TRUE, FALSE)

int_vec <- matrix(c(1, 2, 4, 5), nrow = 2 , ncol = 2)

qt_vec <- rnorm(100000) + 0.2 * g_train_vec[, 1] + 0.3 * g_train_vec[, 1] * g_train_vec[, 4]

res <- train_and_impute_PRS(qt_vec, g_train_vec, g_impute_vec,
       dominance_effects = dom_vec, interaction_effects = int_vec)
```

`var.adj` *Mean and variance effect adjustments.*

Description

Given is a set of (continuous) variables and a qt trait. First, this function adjusts the trait for the mean effects of the variables with a linear model. Next, the variance effect of the variables are estimated and the trait is adjusted further by scaling it in accordance with the results.

Usage

```
var.adj(qt, x, iter_num = 50, eps_param = 1e-10)
```

Arguments

`qt`	A numeric vector.
`x`	A numeric matrix, each column represents a covariate.
`iter_num`	An integer. Represents the number of iterations performed in the Gauss-Newton algorithm
`eps_param`	A number. The Gauss-Newton algorithm terminates if the incriment change of all variance estimates is smaller than this number.

Value

A vector, representing the adjusted trait.

Examples

```
n_val <- 50000
x <- matrix(0,nrow = n_val, ncol = 4)
for(i in 1:4) {
x[, i] <- rnorm(n_val)
}
var_vec <- exp(0.2 * x[, 1] - 0.3 * x[, 4])
qt_vec <- rnorm(n_val, 0, sqrt(var_vec))
res <- var.adj(qt_vec, x)
```

`Var.assoc` *Uncertanty association*

Description

This function finds the association between the predicted uncertanty of some estimates of a trait to the "actual uncertanty" of the estimates. Suppose we have estimates of some trait (this might be a polygenic risk score). Moreover, suppose we have assigned a variance value to each estimate (this might be a variance risk score) to reflect how certain we believe we are about each estimate. Given the true trait values, this function evaluates how well the assigned variance values reflect reality.

We use a likelihood ratio test with 1 degree of freedom * H0: y~N(mu+b*m_score,sigma_sq), * H1: y~N(mu+b*m_score,sigma_sq*(v_score)^a), where y is the trait m_score is the estimate of the trait and v_score is the variance assigned to the estimate. Thus H0 has three degrees of freedom (mu,b,sigma_sq), whereas H1 has four (mu,b,sigma_sq,a)

Usage

```
Var.assoc(qt, m_score, v_score, iter = 50)
```

Arguments

`qt`	A numeric vector.
`m_score`	A numeric vector
`v_score`	A numeric vector with positive values
`iter`	A number of iterations for the Gauss-Newton algorithm

Value

A list with the values:

* a, the association between v_score and the actual variance. * pval, the p-value of the likelihood ratio test

Examples

```
n_val <- 50000L
trait_vec <- rnorm(n_val,0,1)
var_vec <- exp(rnorm(n_val,0,0.1))
est_vec <- trait_vec+rnorm(n_val,0,var_vec)
res <- Var.assoc(trait_vec,est_vec,var_vec, iter = 20)
```

---

var.summary *Variance summary statistics*

Description

Estimates the variance effect of several continuous variables jointly

Usage

```
var.summary(qt, x, iter_num = 50, eps_param = 1e-10)
```

Arguments

qt	A numeric vector.
x	A data frame, each column represents a covariate that should be numeric.
iter_num	An integer. Represents the number of iterations performed in the Gauss-Newton algorithm
eps_param	A number. The Gauss-Newton algorithm terminates if the incriment change of all variance estimates is smaller than this number.

Value

A list with the following objects: * summary, a dataframe with a variance effect estimate for each variable and summary statistics * chi2, the chi2 statistic obtained by considering all parameteres jointly * df, degrees of freedom for the chi2 statistic * pval, p-value of the model * adjusted_values, a vector with qt values that have been adjusted for both mean and variance effects

Examples

```
n_val <- 50000
x <- as.data.frame(matrix(0,nrow = n_val, ncol = 4))
colnames(x) <- c('A','B','C','D')
for(i in 1:4) {
x[, i] <- rnorm(n_val)
}
var_vec <- exp(0.2 * x[, 1] - 0.3 * x[, 4])
qt_vec <- rnorm(n_val, 0, sqrt(var_vec))
res <- var.summary(qt_vec, x)
```

`VarGS.plot` *Actual variance vs predicted variance plot*

Description

This tool creates a line plot that compares the predicted variance of data to its actual variance.

Usage

```
VarGS.plot(
  qt,
  v_score,
  bins = 10,
  xlab = "Predicted variance",
  ylab = "Variance",
  title = ""
)
```

Arguments

`qt`	A numeric vector.
`v_score`	A numeric vector.
`bins`	An integer.
`xlab`	A string.
`ylab`	A string.
`title`	A string.

Value

A plot comparing predicted variance to actual variance.

Examples

```
n_val <- 100000L
v_vec <- exp(rnorm(n_val, 0, 0.1))
qt_vec <- stats::rnorm(n_val, 0, sqrt(v_vec))
VarGS.plot(qt_vec, v_vec)
```

`Viol.by.gen` *Violin plot by genotype*

Description

This tool creates violin plots corresponding to each genotype.

Usage

```
Viol.by.gen(qt, g, trait_name = "qt trait", title = "")
```

Arguments

`qt`	A numeric vector.
`g`	An integer vector.
`trait_name`	A string.
`title`	A string.

Value

A violin plot

Examples

```
n_val <- 50000L
geno_vec <- sample(c(0, 1, 2), size = n_val, replace = TRUE)
qt_vec <- rnorm(n_val) * (1.3^geno_vec) + 1 * geno_vec
Viol.by.gen(qt_vec, geno_vec)
```

Index