Package 'CGEN'

October 13, 2026

Title An R package for analysis of case-control studies in genetic epidemiology

Version 3.48.0

Date 2012-10-10

Description This is a package for analysis of case-control data in genetic epidemiology. It provides a set of statistical methods for evaluating gene-environment (or gene-genes) interactions under multiplicative and additive risk models, with or without assuming gene-environment (or gene-gene) independence in the underlying population.

Imports stats, graphics, utils, grDevices

Depends R (>= 4.0), survival, mvtnorm

License GPL-2 + file LICENSE

Suggests cluster

biocViews SNP, MultipleComparison, Clustering

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

git_branch RELEASE_3_23

git_last_commit 106ed57

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-10-13

Author Samsiddhi Bhattacharjee [aut],
Nilanjan Chatterjee [aut],
Summer Han [aut],
Minsun Song [aut],
William Wheeler [aut],
Matthieu de Rochemonteix [aut],
Nilotpal Sanyal [aut],
Justin Lee [cre]

Maintainer Justin Lee <jhylee@stanford.edu>

Contents

additive.test . . . 2
CGEN . . . 7
getMatchedSets . . . 10
getSummary . . . 12
getWaldTest . . . 13
GxE.scan . . . 14
GxE.scan.combine . . . 16
GxE.scan.partition . . . 17
locusMap.list . . . 19
LocusMapData . . . 20
Manhattan.plot . . . 21
pheno.list . . . 22
printEffects . . . 23
QQ.plot . . . 24
snp.effects . . . 25
snp.effects.plot . . . 27
snp.list . . . 29
snp.logistic . . . 30
snp.matched . . . 34
snp.score . . . 37
subject.list . . . 40
Xdata . . . 41
Xdata2 . . . 41

Index **43**

`additive.test` *A test for gene-environment interaction under an additive risk model for case-control data*

Description

Performs a likelihood ratio test for gene-environment interaction under an additive risk model for case-control data using a standard logistic regression. A set of constraints is imposed to log-odds-ratio parameters to approximate the null model of no interaction under additive risk models. The additive interaction test under gene-environment independence assumption is performed by utilizing the retrospective likelihood by Chatterjee and Carroll (2005).

Usage

```
additive.test(data, response.var, snp.var, exposure.var, main.vars=NULL,
              strata.var=NULL, op=NULL)
```

Arguments

`data`	Data frame containing all the data. No default.
`response.var`	Name of the binary response variable coded as 0 (controls) and 1 (cases). No default.
`snp.var`	Name of the genotype variable coded as 0, 1, 2 (or 0, 1). No default.
`exposure.var`	Name of the exposure variable coded as 0, 1, 2 (or 0, 1). No default.
`main.vars`	Character vector of variable names or a formula for all covariates of interest which need to be included in the model as main effects. Do not include snp.var and exposure.var here, as their main effects will automatically be included in the model. The default is NULL.
`strata.var`	For a retrospective likelihood, name of the stratification variable, such as race, ethnicity, geographic origin, or principal components of population stratification markers (Bhattacharjee et al. 2010 AJHG). It is highly recommended to include it if present. This option takes into account gene-environment dependence due to population stratification by allowing the genotype frequency to vary by the discrete level of the stratification variable. The default is NULL.
`op`	A list of options with possible names `genetic.model`, `optim.method`, `indep`, `maxiter` and `reltol` (see details). The default is NULL.

Details

A maximum likelihood for a full model is obtained by optimizing a logistic regression model using a standard binomial likelihood (i.e. prospective likelihood) while a maximum likelihood for a null model is obtained by fitting a reduced model with a set of constaints imposed on logistic regression parameters to approximate the null model of no interaction in an additive risk model. The additive interaction test under the gene-environment independence assumption can be conducted by utilizing the retrospective likelihood by Chatterjee and Carroll (2005). The following is the definition of the likelihood under the gene-environment independence assumption:

Definition of the likelihood under the gene-environment independence assumption:

Let D = 0, 1 be the case-control status, G = 0, 1, 2 denote the SNP genotype, S = 1, ..., k denote the levels of the stratification variable and Z be the design matrix for all the covariates including G, the interactions, and a column for the intercept parameter. If f_s denotes the allele frequency for stratum s, then

$$P(G = 0) = (1 - f_s)^2$$

$$P(G = 1) = 2f_s(1 - f_s)$$

$$P(G = 2) = f_s^2.$$

If $\xi_s = \log(f_s/(1 - f_s))$, then

$$\log\left(\frac{P(G = 1)}{P(G = 0)}\right) = \log(2) + \xi_s$$

and

$$\log\left(\frac{P(G = 2)}{P(G = 0)}\right) = 2\xi_s$$

Let $\theta(d,g) = d * Z * \beta + I(g=1) * \log(2) + g * \xi_s$.

Then the likelihood for a subject is $P(D=d, G=g|Z,S) = \frac{\exp(\theta(d,g))}{\sum_{d,g}\exp(\theta(d,g))}$ where the sum is taken over the 6 combinations of d and g.

Options list:

Below are the names for the options list `op`. All names have default values if they are not specified.

- `genetic.model` 1-3 where 1=dominant, 2=recessive, 3=general. The default is 3.
- `optim.method` One of "BFGS", "CG", "L-BFGS-B", "Nelder-Mead", "SANN". The default is "BFGS".
- `indep` TRUE for using a retrospective likelihood for gene-environment independence assumption. FALSE for using a standard prospective likelihood. The default is FALSE.
- `reltol` Stopping tolerance. The default is 1e-7.
- `maxiter` Maximum number of iterations. The default is 500.

Value

A list containing the following:

- additive
  - `pval.add.LRT` P-value of the additive interaction likelihood ratio test.
  - `pval.add.UML` P-value of the additive interaction UML Wald test.
  - `pval.add.CML` (only for indep=T) P-value of the additive interaction CML Wald test.
  - `pval.add.EB` (only for indep=T) P-value of the additive interaction EB Wald test.
  - `LRT.add` Likelihood ratio test statistic for the additive interaction.
  - `RERI.UML` UML estimate of RERI.
  - `RERI.CML` (only for indep=T) CML estimate of RERI.
  - `RERI.EB` (only for indep=T) EB estimate of RERI.
  - `RERI.test` (only for 2x2 method) Test based on RERI.
  - `S.test` (only for 2x2 method and indep=F) Test based on Synergy Index.
  - `AP.test` (only for 2x2 method and indep=F) Test based on "Attributable Proportion due to interaction".
- multiplicative (only for genetic.model != 0)
  - `pval.mult.LRT` P-value of the multiplicative interaction likelihood ratio test.
  - `pval.mult.UML` P-value of the multiplicative interaction UML Wald test.
  - `pval.mult.CML` (only for indep=T) P-value of the multiplicative interaction CML Wald test.
  - `pval.mult.EB` (only for indep=T) P-value of the multiplicative interaction EB Wald test.
  - `LRT.mult` Likelihood ratio test statistic for the multiplicative interaction.
- model.info
  - `response.var` Response variable name
  - `snp.var` SNP variable name

- `exposure.var` Exposure variable name
- `main.vars` Names of covariates which are included in the model as main effects. This does not include snp.var and exposure.var, whose main effects are included in the model by default.
- `strata.var` Name of the stratification variable for a retrospective likelihood.
- `op` List of options used in the fit; includes genetic.model, optim.method, indep, maxiter, reltol
- `GxEtable` The frequency table defined by table(snp.var, exposure.var)
- `method` One of "2x2", "2x3", "3x2", "3x3", "3x2trend" ("3x3trend" is currently not available)
- `parms.lm.UML` UML parameters estimates
- `parms.lm.CML` (only for indep=T) CML parameters estimates
- `parms.lm.EB` (only for indep=T) EB parameters estimates
- `cov.lm.UML` UML parameters covariance matrix estimate
- `cov.lm.CML` (only for indep=T) CML parameters covariance matrix estimate
- `cov.lm.EB` (only for indep=T) EB parameters covariance matrix estimate
- `full.loglike.UML` (only for indep=T) Loglikelihood of UML full model fit
- `full.loglike.CML` (only for indep=T) Loglikelihood of CML full model fit
- `DF` Degrees of freedom of the additive/multiplicative interaction test.
- `optim.out` (only for genetic.model != 0) Optimization output of the optim() function for a null model under an additive model restriction.
- `OR.table.UML` (only for indep=F) Odds ratio table UML estimate
- `OR.table.CML` (only for indep=T) Odds ratio table CML estimate

References

Sanyal, N., Napolioni, V., de Rochemonteix, M., Belloy, M. E., Caporaso, N. E., Landi, M. T., Greicius, M. D., Chatterjee, N., Han, S. S. A robust test for additive gene-environment interaction under the trend effect of genotype using an empirical Bayes-type shrinkage estimator. American Journal of Epidemiology 2021 (in press).

de Rochemonteix M, Napolioni V, Sanyal N, et al. A likelihood ratio test for gene-environment interaction based on the trend effect of genotype under an additive risk model using the gene-environment independence assumption. American Journal of Epidemiology 2021;190(1):129-41.

Han, S. S, Rosenberg P. S, Garcia-Closas M, Figueroa J. D, Silverman D, Chanock S. J, Rothman N, and Chatterjee N. Likelihood ratio test for detecting gene (G) environment (E) interactions under the additive risk model exploiting G-E independence for case-control data. Am J of Epidemiol, 2012; 176:1060-7.

Chatterjee, N. and Carroll, R. Semiparametric maximum likelihood estimation exploting gene-environment independence in case-control studies. Biometrika, 2005, 92, 2, pp.399-418.

Bhattacharjee, S., Wang, Z., Ciampa, J., Kraft, P., Chanock, S., Yu, K. and Chatterjee, N. Using principal components of genetic variation for robust and powerful detection of gene-gene interactions in case-control and case-only studies. Am J of Hum Genet, 2010; 86(3), 331-342.

See Also

`snp.logistic`, `snp.score`

Examples

```
# # # EXAMPLE 1: Use the lung cancer data
data(Xdata2, package="CGEN")

# <<1A>> Standard (prospective) likelihood additive GXE analysis under the additive genetic model
# (i.e. the trend effect of genotype) using the method from de Rochemonteix et al. 2021)

A0F <- additive.test(data = Xdata2, response.var = "case.control", snp.var = "SNP",
          exposure.var = "smoking", main.vars = c("cov1", "cov2", "cov3",
          "cov4", "study"), strata.var = "study", op = list(genetic.model = 0))
A0F$pval.add.LRT   # P-value of the additive interaction likelihood ratio test.

# <<1B>> Retrospective likelihood additive GXE analysis under the additive genetic model
# (i.e. the trend effect of genotype) using the method (LRT) from de Rochemonteix et al. 2021
# and, for the empirical-Bayes type shrinkage estimator, the method (Wald test) from
# Sanyal et al. 2021)

A0T <- additive.test(data = Xdata2, response.var = "case.control", snp.var = "SNP",
          exposure.var = "smoking", main.vars = c("cov1", "cov2", "cov3",
          "cov4", "study"), strata.var = "study", op = list(genetic.model = 0,
          indep = TRUE))
A0T$pval.add.LRT   # P-value of the additive interaction likelihood ratio test.
A0T$pval.add.EB    # P-value of the additive interaction EB Wald test
                   #(that is the weighted average of the CML and UML methods.)
A0T$pval.add.UML   # P-value of the additive interaction UML Wald test
                   #(that does not assume G-E independence).
A0T$pval.add.CML   # P-value of the additive interaction CML Wald test
                   #(that assumes G-E independence).

# <<1C>> Standard (prospective) likelihood additive GXE analysis under the dominant genetic model
# (using the method from Han et al. 2012)

A1F <- additive.test(data = Xdata2, response.var = "case.control", snp.var = "SNP",
          exposure.var = "smoking", main.vars = c("cov1", "cov2", "cov3",
          "cov4", "study"), strata.var = "study", op = list(genetic.model = 1))
A1F$pval.add.LRT   # P-value of the additive interaction likelihood ratio test.

# <<1D>> Standard (prospective) likelihood additive GXE analysis under the dominant genetic model
# (using the methofd from Han et al. 2012)

A1T <- additive.test(data = Xdata2, response.var = "case.control", snp.var = "SNP",
          exposure.var = "smoking", main.vars = c("cov1", "cov2", "cov3",
          "cov4", "study"), strata.var = "study", op = list(genetic.model = 1,
          indep = TRUE))
A1T$pval.add.LRT   # P-value of the additive interaction likelihood ratio test.



# # # EXAMPLE 2: Use the ovarian cancer data
data(Xdata, package="CGEN")

table(Xdata[, "gynSurgery.history"])
```

```
# Recode the exposure variable so that it is 0-1

temp <- Xdata[, "gynSurgery.history"] == 2
Xdata[temp, "gynSurgery.history"] <- 1

# <<2A>> Standard (prospective) likelihood additive GXE analysis under the dominant genetic model
# (using the method from Han et al. 2012)

out1 <- additive.test(data=Xdata, response.var="case.control", snp.var="BRCA.status",
           exposure.var="gynSurgery.history", main.vars=c("n.children","oral.years"),
           op=list(genetic.model=1))

# <<2B>> Retrospective likelihood (indep = TRUE) additive GXE analysis under the dominant genetic model
# (using the method from Han et al. 2012)

out2 <- additive.test(data=Xdata, response.var="case.control", snp.var="BRCA.status",
           exposure.var="gynSurgery.history", main.vars=~n.children+oral.years,
           strata.var="ethnic.group", op=list(indep=TRUE, genetic.model=1))
```

CGEN *An R package for analysis of case-control studies in genetic epidemiology*

Description

The new version extends the additive.test() function by implementing additive gene-gene or gene-environment tests under the trend effect of genotypes (i.e. additive genetic model), extending the previous approaches that were only available for the dominant or recessive genetic models (Han et al. 2012 AJE). Several types of interaction tests under the trend effect of genotype are available – methods with and without the gene-environment independence assumption (de Rochemonteix et al. 2020 AJE) and robust methods based on an empirical Bayes type shrinkage estimator (Sanyal et al. 2021 AJE).

This package is for logistic regression analyses of SNP data in case-control studies. It is designed to give the users flexibility of using a number of different methods for analysis of SNP-environment or SNP-SNP interactions. It is known that power of interaction analysis in case-control studies can be greatly enhanced if it can be assumed that the factors (e.g. two SNPs) under study are independently distributed in the underlying population. The package implements a number of different methods that can incorporate such independence constraints into analysis of interactions in the setting of both unmatched and matched case-control studies. These methods are more general and flexible than the popular case-only method of analysis of interaction that also assumes gene-gene or/and gene-environment independence for the underlying factors in the underlying population. The package also implements various methods, based on shrinkage estimation and conditional-likelihoods, that can automatically adjust for possible violation of the independence assumption that could arise due to direct causal relationship (e.g. between a gene and a behavior exposure) or indirect correlation (e.g due to population stratification). A number of convenient summary and printing functions are included. In its previous version, the package has been updated with the new function `snp.score` that allow testing for disease-SNP association accounting for gene-environment interaction using an

array of different types of score-tests. The function can handle both genotyped and imputed SNPs. The latest version of the package has extended the additive.test() function by implementing additive gene by environment tests under the trend effect of genotypes (additive model). Two types of trend tests are available - likelihood ratio tests and Wald tests based on an empirical Bayes type shrinkage estimator. The package will continue to be updated with new methods as they are developed. The methods are currently not suitable for analysis of SNPs on sex chromosomes.

Details

The main functions for unmatched data are `additive.test`, `snp.logistic` and `snp.score`. Whereas `additive.test`, `snp.logistic` and `snp.score` analyzes one SNP with each function call, `GxE.scan` analyzes a collection of SNPs and writes the summary results to an external file. With `additive.test` and `snp.logistic`, a data frame is input in which the SNP variable must be coded as 0-1-2 (or 0-1). The function `snp.score` can be used with imputed genotypes, where the SNP variable is coded as the expected dosage. The functions `getSummary`, `getWaldTest` and `snp.effects` can be called for creating summary tables, computing Wald tests and joint/stratified effects using the returned object from `snp.logistic` (see `Examples` in `snp.logistic`). With `GxE.scan`, the data is read in from external files defined in `snp.list` and `pheno.list`. The collection of p-values computed in `GxE.scan`, can be plotted using the functions `QQ.plot` and `Manhattan.plot`.
The function for analysis of matched case-control data is `snp.matched`. Optimal matching can be obtained from the function `getMatchedSets`. The current version of the package is only suitable for analysis of SNPs on non-sex chromosomes.

Main functions for single SNP analysis:
`additive.test`
`snp.logistic`
`snp.matched`
`snp.score`

For GWAS analysis:
`GxE.scan`
`GxE.scan.combine`
`GxE.scan.partition`
`pheno.list`
`snp.list`
`subject.list`

Graphics:
`Manhattan.plot`
`QQ.plot`
`snp.effects.plot`

Sample data:
`LocusMapData`
`Xdata`

Miscellaneous:

getMatchedSets (Used with snp.matched)
getSummary (The same as calling summary)
getWaldTest (For computing Wald tests)
locusMap.list (Used with Manhattan.plot)
printEffects (The same as calling print)
snp.effects (For computing joint and stratified effects)

Author(s)

Samsiddhi Bhattacharjee, Summer Han, Minsun Song, Nilanjan Chatterjee, William Wheeler and Nilotpal Sanyal <nsanyal@stanford.edu>

References

Maximum-likelihood estimation under independence

Chatterjee, N. and Carroll, R. Semiparametric maximum likelihood estimation exploting gene-environment independence in case-control studies. Biometrika, 2005, 92, 2, pp.399-418.

Shrinkage estimation

Mukherjee B, Chatterjee N. Exploiting gene-environment independence in analysis of case-control studies: An empirical Bayes approach to trade-off between bias and efficiency. Biometrics 2008, 64(3):685-94.

Mukherjee B et al. Tests for gene-environment interaction from case-control data: a novel study of type I error, power and designs. Genetic Epidemiology, 2008, 32:615-26.

Chen YH, Chatterjee N, Carroll R. Shrinkage estimators for robust and efficient inference in haplotype-based case-control studies. Journal of the American Statistical Association, 2009, 104: 220-233.

Conditional Logistic Regression and Adjustment for Population stratification

Chatterjee N, Zeynep K and Carroll R. Exploiting gene-environmentindependence in family-based case-control studies: Increased power for detecting associations, interactions and joint-effects. Genetic Epidemiology2005; 28:138-156.

Bhattacharjee S, Wang Z, Ciampa J, Kraft P, Chanock S, Yu K, Chatterjee N Using Principal Components of Genetic Variation for Robust and Powerful Detection of Gene-Gene Interactions in Case-Control and Case-Only studies. American Journal of Human Genetics, 2010, 86(3):331-342.

Score tests

Han, S.S., Rosenberg, P., Ghosh, A., Landi M.T., Caporaso N. and Chatterjee, N. An exposure weighted score test for genetic association integrating environmental risk-factors. Biometrics 2015 (Article first published online: 1 JUL 2015 | DOI: 10.1111/biom.12328)

Song M., Wheeler B., Chatterjee, N. Using imputed genotype data in joint score tests for genetic association and gene-environment interactions in case-control studies (In preparation).

Tests for additive interaction

Sanyal, N., Napolioni, V., de Rochemonteix, M., Belloy, M. E., Caporaso, N. E., Landi, M. T., Greicius, M. D., Chatterjee, N., Han, S. S. A robust test for additive gene-environment interaction under the trend effect of genotype using an empirical Bayes-type shrinkage estimator. American Journal of Epidemiology 2021 (in press).

de Rochemonteix M, Napolioni V, Sanyal N, et al. A likelihood ratio test for gene-environment interaction based on the trend effect of genotype under an additive risk model using the gene-environment independence assumption. American Journal of Epidemiology 2021;190(1):129-41.

Han, S. S, Rosenberg P. S, Garcia-Closas M, Figueroa J. D, Silverman D, Chanock S. J, Rothman N, and Chatterjee N. Likelihood ratio test for detecting gene (G) environment (E) interactions under the additive risk model exploiting G-E independence for case-control data. Am J of Epidemiol, 2012; 176:1060-7.

`getMatchedSets` *Case-Control and Nearest-Neighbor Matching*

Description

Obtain matching of subjects based on a set of covariates (e.g., principal components of population stratification markers). Two types of matcing are allowed 1) Case-Control(CC) matching and/or 2) Nearest-Neighbour(NN) matching.

Usage

```
getMatchedSets(x, CC, NN, ccs.var=NULL, dist.vars=NULL, strata.var=NULL,
            size=2, ratio=1, fixed=FALSE)
```

Arguments

`x`	Either a data frame containing variables to be used for matching, or an object returned by `dist` or `daisy` or a matrix coercible to class dist. No default.
`CC`	Logical. TRUE if case-control matching should be computed, FALSE otherwise. No default.
`NN`	Logical. TRUE if nearest-neighbor matching should be computed, FALSE otherwise. No default. At least one of CC and NN should be TRUE.
`ccs.var`	Variable name, variable number, or a vector for the case-control status. If `x` is dist object, a vector of length same as number of subjects in `x`. This must be specified if CC=TRUE. The default is NULL.
`dist.vars`	Variables numbers or names for computing a distance matrix based on which matching will be performed. Must be specified if `x` is a data frame. Ignored if `x` is a distance. Default is NULL.
`strata.var`	Optional stratification variable (such as study center) for matching within strata. A vector of mode integer or factor if `x` is a distance. If `x` is a data frame, a variable name or number is allowed. The default is NULL.
`size`	Exact size or maximum allowable size of a matched set. This can be an integer greater than 1, or a vector of such integers that is constant within each level of `strata.var`. The default is 2.
`ratio`	Ratio of cases to controls for CC matching. Currently ignored if fixed = FALSE. This can be a positive number, or a numeric vector that is constant within each level of `strata.var`. The default is 1.
`fixed`	Logical. TRUE if "size" should be interpreted as "exact size" and FALSE if it gives "maximal size" of matched sets. The default is FALSE.

Details

If a data frame and dist.vars is provided, dist along with the euclidean metric is used to compute distances assuming conituous variables. For categorical, ordinal or mixed variables using a custom distance matrix such as that from daisy is recommended. If strata.var is provided both case-control (CC) and nearest-neighbor (NN) matching are performed within strata. size can be any integer greater than 1 but currently the matching obtained is usable in snp.matched only if size is 8 or smaller, due to memory and speed limitations.

When fixed=FALSE, NN matching is computed using a modified version of hclust, where clusters are not allowed to grow beyond the specified size. CC matching is computed similarly with the further constraint that each cluster must have at least one case and one control. Clusters are then split up into 1:k or k:1 matched sets, where k is at most size - 1 (known as full matching). For exactly optimal full matching use package optmatch.

When fixed=TRUE, both CC and NN use heuristic fixed-size clustering algorithms. These algorithms start with matches in the periphery of the data space and proceed inward. Hence prior removal of outliers is recommended. For CC matching, number of cases in each matched set is obtained by rounding size * [ratio/(1+ratio)] to the nearest integer. The matching algorithms for fixed=TRUE are faster, but in case of CC matching large number of case or controls may be discarded with this option.

Value

A list with names "CC", "tblCC", "NN", and "tblNN". "CC" and "NN" are vectors of integer labels defining the matched sets, "tblCC" and "tblNN" are matrices summarizing the size distribution of matched sets across strata. i'th row corresponds to matched set size of i and columns represent different strata. The order of strata in columns may be different from that in strata.var, if strata.var was not coded as successive integers starting from 1.

References

Luca et al. On the use of general control samples for genome-wide association studies: genetic matching highlights causal variants. Amer Jour Hum Genet, 2008, 82(2):453-63.

Bhattacharjee S, Wang Z, Ciampa J, Kraft P, Chanock S, Yu K, Chatterjee N. Using Principal Components of Genetic Variation for Robust and Powerful Detection of Gene-Gene Interactions in Case-Control and Case-Only studies. American Journal of Human Genetics, 2010, 86(3):331-342.

See Also

snp.matched

Examples

```
# Use the ovarian cancer data
 data(Xdata, package="CGEN")

# Add fake principal component columns.
```

```
# set.seed(123)
Xdata <- cbind(Xdata, PC1 = rnorm(nrow(Xdata)), PC2 = rnorm(nrow(Xdata)))

# Assign matched set size and case/control ratio stratifying by ethnic group
size <- ifelse(Xdata$ethnic.group == 3, 2, 4)
ratio <- sapply(Xdata$ethnic.group, switch, 1/2 , 2 , 1)
mx <- getMatchedSets(Xdata, CC=TRUE, NN=TRUE, ccs.var="case.control",
                     dist.vars=c("PC1","PC2") , strata.var="ethnic.group",
       size = size, ratio = ratio, fixed=TRUE)
mx$NN[1:10]
mx$tblNN

# Example of using a dissimilarity matrix using catergorical covariates with
#  Gower's distance
library("cluster")
d <- daisy(Xdata[, c("age.group","BRCA.history","gynSurgery.history")] ,
           metric = "gower")
# Specify size = 4 as maximum matched set size in all strata
mx <- getMatchedSets(d, CC = TRUE, NN = TRUE, ccs.var = Xdata$case.control,
                     strata.var = Xdata$ethnic.group, size = 4,
       fixed = FALSE)
mx$CC[1:10]
mx$tblCC
```

`getSummary` *Compute summary information*

Description

Returns a matrix of estimated parameters, standard errors, test statistics, and p-values.

Usage

```
getSummary(fit, sided=2, method=NULL)
```

Arguments

`fit`	The return object from `snp.logistic`, `snp.matched`, `glm()`, or a list with names "parms" and "cov" containing parameter estimates and the variance-covariance matrix for the estimates. No default.
`sided`	1 or 2 for a 1 or 2 sided p-values. The default is 2.
`method`	Vector of values from "UML", "CML", "EB" or "CCL", "HCL", "CLR". The default is NULL.

Details

This function returns a matrix similar to `summary(glm.obj)$coefficients`, except the p-values are always computed using the normal distribution.

Value

A matrix with column names "Estimate", "Std.Error", "Z.value", and "Pvalue". The rownames of the returned matrix will be the names of `parms` if `parms` is a vector.

Examples

```
parms <- 1:5
cov   <- matrix(data=1, nrow=5, ncol=5)
getSummary(list(parms=parms, cov=cov))

# Compare to summary()
# set.seed(123)
n <- 100
y <- rbinom(n, 1, 0.5)
x <- cbind(runif(n), rbinom(n, 1, 0.5))
fit <- glm(y ~ x, family=binomial())
sum <- summary(fit)
sum$coefficients
getSummary(fit)
```

---

getWaldTest *Compute a Wald test*

Description

Computes a univariate or multivariate Wald test

Usage

```
getWaldTest(fit, parmNames, method=NULL)
```

Arguments

`fit`	Return object from `snp.logistic`, `snp.matched`, `glm()` or a list with names "parms" and "cov" (see details). No default.
`parmNames`	Vector of parameters to test. This vector can be a character vector of parameter names or a numeric vector of positions. No default.
`method`	Vector of values from "UML", "CML", "EB" or "CCL", "HCL", "CLR". The default is NULL.

Details

If `fit` is a list, then "parms" should be the vector of coefficients, and "cov" should be the covariance matrix. If `parmNames` is a character vector, then "parms" should be a named vector and the names must match the rownames and colnames of "cov". A chi-squared test is computed.

Value

List containing the value of the test statistic (`test`), degrees of freedom (`df`), and p-value (`pvalue`).

Examples

```
# set.seed(123)
n <- 100
y <- rbinom(n, 1, 0.5)
x <- runif(n*5)
dim(x) <- c(n, 5)
x <- data.frame(x)
colnames(x) <- c("x", "x2", "x3", "z", "z2")
fit <- glm(y ~ ., data=x, family=binomial())

# Chi-squared test
getWaldTest(fit, c("x", "z"))

beta <- c(-2.5, 2.5)
cov  <- diag(1:2)
getWaldTest(list(parms=beta, cov=cov), 1:2)
```

`GxE.scan` *GxE analysis for an array of SNPs*

Description

Performs analyses for a scan of SNPs.

Usage

```
GxE.scan(snp.list, pheno.list, op=NULL)
```

Arguments

`snp.list`	See `snp.list`. No default.
`pheno.list`	See `pheno.list`. No default.
`op`	See details for this list of options. The default is NULL.

Details

For proper anlaysis of imputed SNPs, the only option that should be used is `model`=3 (`snp.score`)

See the vignette `vignette_GxE` for examples of running this function. To use this function, the data must be stored in files as defined in `snp.list` and `pheno.list`. See the examples on how to create these lists. The format of the genotype data must be a format where the SNPs are rows (`snp.list$format`="impute", "ldat", "tped") or a format where either the GLU software or PLINK software can read and transform the data. Options for setting the path to GLU or PLINK are found

in snp.list. The genotype data is read in from the file snp.list$file, and the variables for the main effects and interactions are read in from the file pheno.list$file. The subjects to be included in the analysis are the subjects with matching ids in the phenotype and genotype data.

Users can easily run their own customized scan by setting the option model to 0 and setting the option scan.func to their own scan function. The vignette "vignette_GxE" has examples of user-defined scans.

Options list op: Below are the names for the options list op. All names have default values if they are not specified.

- model 0-4, where 1 = snp.logistic, 2 = additive.test, 3 = snp.score, 4 = snp.matched and 0 is for a user defined function. The default is 1.
- out.file File name to save the results. If NULL, then the output file will be created in the working directory as paste(getwd(), "/GxE.scan.output.txt", sep="").
- scan.func.op List of specific options for the scan function. For model=1-4, see snp.logistic, additive.test, snp.score or snp.matched for these options. The default is NULL.
- geno.counts 0 or 1 to write the genotype frequency counts to the output file. The default is 1.
- geno.MAF 0 or 1 to write the SNP MAF to the output file. The default is 1.
- geno.missRate 0 or 1 to write the SNP missing rate to the output file. The default is 1.
- output.methods Character vector of "UML", "CML", "EB" to only output results for specific methods. This option is only for snp.logistic and snp.score. The default is c("UML", "CML", "EB").
- output.parms 0 or 1 to output parameter estimates. This option is only for snp.logistic and snp.score. The default is 0.
- output.sandwich 0 or 1 to output sandwich covariance estimates. This option is only for snp.score. The default is 0.

Advanced options:

- scan.func (For model = 0). The name of the user-defined scan function. This function must have 2 input arguments and return a named list or named vector. See the vignette "vignette_GxE" for examples. The default is NULL.
- scan.setup.func (For model = 0). NULL or the name of the user-defined function called after the phenotype data is read in and before the genotype data is read. This function is not required for model = 0. This function must have 2 input arguments and return NULL or a named list. See the vignette "vignette_GxE" for examples. The default is NULL.

Output variable names: The output file will contain columns for the SNP, MAF, missing rate, and genotype frequency counts. Other columns are specific for the specified model. A column denoted by *.Inter.Pvalue is a p-value from a Wald test involving the interaction terms of the SNP and exposure variables. A column denoted by *.Omnibus.Pvalue is a p-value from a Wald test involving the main effect of the SNP and interaction terms of the SNP and exposure variables.

Value

The name of the output file containing the results. See the option `out.file`. See details for the column names in the output file.

See Also

`GxE.scan.partition`, `GxE.scan.combine`

Examples

```
# Define the list for the genotype data.
snp.list <- list()
snp.list$file <- system.file("sampleData", "geno_data.ldat.gz", package="CGEN")
snp.list$file.type <- 7
snp.list$delimiter <- "\t"
snp.list$in.miss <- "  "

# Only process the first 5 SNPs in the file
snp.list$start.vec <- 1
snp.list$stop.vec <- 6

# Define pheno.list
pheno.list <- list()
pheno.list$file <- system.file("sampleData", "Xdata.txt", package="CGEN")
pheno.list$file.type <- 3
pheno.list$delimiter <- "\t"
pheno.list$id.var <- "id"

# Define the variables in the model
pheno.list$response.var <- "case.control"
pheno.list$strata.var <- "ethnic.group"
pheno.list$main.vars <- c("age.group", "oral.years", "n.children")
pheno.list$int.vars <- "n.children"

# Define the list of options
op <- list(out.file="out.txt")

# For this model, all variables are continuous
# temp <- GxE.scan(snp.list, pheno.list, op=op)
```

GxE.scan.combine *Combines output files into one file*

Description

Combines the output files from running GxE.scan on a cluster.

Usage

```
GxE.scan.combine(out.file, dir.or.files, pattern="GxEout_")
```

Arguments

out.file	Name of the output file of combined results. No default.
dir.or.files	Directory containing the output files from GxE.scan or the full names of the files to be combined. If length(dir.or.files) = 1, then it is assumed to be a directory. No default.
pattern	Character string to match file names when dir.or.files is a directory. The default is "GxEout_".

Details

This function would be called after all the submitted jobs from GxE.scan.partition have finished running. It is assumed that the files to be combined all have a header of variable names.

Value

A character vector of the files combined.

See Also

GxE.scan, GxE.scan.partition

Examples

```
out.file <- "/data/results/all_output.txt"
dir      <- "/data/out"
# GxE.scan.combine(out.file, dir)
```

GxE.scan.partition *Creates GxE.scan job files for a computing cluster*

Description

Creates job files for running GxE.scan on a parallel processing system.

Usage

```
GxE.scan.partition(snp.list, pheno.list, op=NULL)
```

Arguments

snp.list	See snp.list and details below. No default.
pheno.list	See pheno.list. No default.
op	See details for this list of options. The default is NULL.

Details

This function will create files needed for running a GWAS scan on a computing cluster. The user must know how to submit jobs and know how to use their particular cluster. On many clusters, the command for submitting a job is "qsub". The scan is partitioned into smaller jobs by either setting the values for `snp.list$start.vec` and `snp.list$stop.vec` or by setting the value for `snp.list$include.snps`. The partitioning is done so that each job will process an equal number of SNPs. In the output directory (see option `out.dir`), three types of files will be created. One type of file will be the R program file containing R statements defining `snp.list`, `pheno.list` and `op` for the `GxE.scan` function. These files have the ".R" file extension. Another type of file will be the job file which calls the R program file. These files are named `paste(op$out.dir, "job_", op$id.str, 1:op$n.jobs, sep="")` The third type of file is a single file containing the names of all the job files. This file has the prefix "Rjobs_". This function will automatically set the name of the output file created by `GxE.scan` to a file in the `op$out.dir` directory with the prefix "GxEout_".

Options list op: Below are the names for the options list `op`. All names have default values if they are not specified.

- `n.jobs` The (maximum) number of jobs to run. The default is 100.
- `out.dir` Directory to save all files. If NULL, then the files will be created in the working directory `getwd`.
- `GxE.scan.op` List of options for the `GxE.scan` function. The default is NULL.
- `R.cmd` Character string for calling R. The default is "R –vanilla".
- `begin.commands.R` Character vector of R statements to be placed at the top of each R program file. For example, `begin.commands.R`=c("rm(list=ls(all=TRUE))", "gc()", 'library(CGEN, lib.loc="/home/Rlibs/")') The default is "library(CGEN)".
- `qsub.cmd` Character string for the command to submit a single job. The default is "qsub".
- `begin.commands.qsub` Character vector of statements to be placed at the top of each job file. For example, `begin.commands.qsub`="module load R". The default is NULL.
- `id.str` A character string to be appended to the file names. The default is "".

snp.list
The objects `start.vec` and `stop.vec` in `snp.list` are set automatically, so they do not need to be set by the user. In general, it is more efficient in terms of memory usage and speed to have the genotype data partitioned into many files. Thus, `snp.list$file` can not only be set to a single file but also set to a character vector of the partitioned files when calling this function. In this case, the number of jobs to create (`op$n.jobs`) must be greater than or equal to the number of partitioned files. An object in `snp.list` that is unique to the `GxE.scan.partition` function is `nsnps.vec`. Each element of `snp.list$nsnps.vec` is the number of SNPs in each file of `snp.list$file`. If `nsnps.vec` is not specified and `snp.list$file` contains more than one file, then each job will process an entire file in `snp.list$file`.

For the scenarios when the genotype data must be transformed and the data is contained in a single file, then `snp.list$include.snps` should also be set. This will create a separate list of SNPs for each job to process.

Value

The name of the file containing names of the job files to be submitted. See details.

See Also

GxE.scan, GxE.scan.combine

Examples

```
# Define the list for the genotype data. There are 50 SNPs in the TPED file.
snp.list <- list(nsnps.vec=50, format="tped")
snp.list$file <- system.file("sampleData", "geno_data.tped.gz", package="CGEN")
snp.list$subject.list <- system.file("sampleData", "geno_data.tfam", package="CGEN")

# Define pheno.list
pheno.list <- list(id.var=c("Family", "Subject"), delimiter="\t", header=1,
                   response.var="CaseControl")
pheno.list$file <- system.file("sampleData", "pheno.txt", package="CGEN")
pheno.list$main.vars <- ~Gender + Exposure
pheno.list$int.vars <- ~Exposure
pheno.list$strata.var <- "Study"

# Define the list of options.
# Specifying n.jobs=5 will let each job process 10 SNPs.
op <- list(n.jobs=5, GxE.scan.op=list(model=1))

# GxE.scan.partition(snp.list, pheno.list, op=op)
```

---

locusMap.list *List to describe the locus map data*

Description

The list to describe the locus map data for Manhattan.plot.

Format

The format is: List of 8

file File containing the locus map data. This file must contain at least three columns: a column for the SNP names, a column for the chromosomes, and a column for the location of the SNP on the chromosome. The location should be numeric values. No default.

file.type 1, 3 or 4 (see details). The default is 3.

delimiter The delimiter used in the files. The default is "\t" (a tab).

header 0 or 1 if the file contains a header of variable names. The default is 0.

snp.var Variable name (e.g. rs number) or column number of the SNP (locus) variable. No default.

chrm.var Variable name (e.g. chromosome number) or column number of the chromosome variable. No default.

loc.var Variable name or column number of the location variable, which denotes the SNP's position on the chromosome. This variable should be numeric. No default.

Details

In this list, `file` must be specified. The types of files are described below.

- `Type 1` An .rda file where the saved object was a data frame.
- `Type 3` A delimited file.

`LocusMapData` *Locus map data*

Description

Locus map data for `Manhattan.plot`

Details

LocusMapData.txt is a tab delimited file that contains sample chromosome and location information. The first 5 rows look like:

SNP	CHROMOSOME	LOCATION
rs11102647	1	113783261
rs6695241	1	172626514
rs12567796	1	18262009
rs2810583	1	41549436

Examples

```
# Load and print the first 5 rows
data(LocusMapData, package="CGEN")

LocusMapData[1:5, ]
```

Manhattan.plot *Manhattan plot*

Description

Creates a Manhattan plot

Usage

```
Manhattan.plot(infile, plot.vars, locusMap.list, op=NULL)
```

Arguments

infile	Output file from GxE.scan. No default.
plot.vars	Character vector of the variables in infile to plot. These variables should p-values. No default.
locusMap.list	See locusMap.list. No default.
op	List of options (see details). The default is NULL.

Details

Plots p-values on a minus log base 10 scale versus the locations of the SNPs on each chromosome.

Options list op: Below are the names for the options list op. All names have default values if they are not specified.

- add A number to add spacing between the chromosomes. The default is 0.
- cex Magnification factor for the plotting symbol. The default is 1.
- cex.axis Magnification factor for the axes. The default is 1.
- cex.lab Magnification factor for the axes labels. The default is 1.
- cex.main Magnification factor for the title. The default is 1.
- colors Character vector of colors to use in the plot. See colors for all possible colors. The default is NULL.
- figs Two element vector for the number of rows and columns if length(plot.vars) > 1 and onePlot = 0. The default is NULL.
- min.p All p-values less than min.p will be set to min.p. The default is 1e-30.
- onePlot 0 or 1 to plot all of plot.vars on the same screen. The default is 0.
- pch Vector of plotting symbols to use. See points for the different plotting symbols.
- splitScreen 0 or 1 to split the plot into two seperate parts. The default is 0.
- subset Vector of chromosomes to plot. The default is NULL.
- tcl The length of tick marks as a fraction of the height of a line of text. The default is -0.5.
- x.las 0-3 for axis labels. 0=parallel, 1=horizontal, 2=perpendicular, 3=vertical The default is 2.
- yaxis.range Vector of length 2 to set the limits for the y-axis. The limits should be on the original scale. The default is NULL.

Value

Returns nothing. Creates a Manhattan plot.

See Also

QQ.plot, locusMap.list

Examples

```
# Load the data containing the chromosomes and locations
data(LocusMapData, package="CGEN")

# For illustrative purposes, add some hypothetical p-values to x
# set.seed(123)
LocusMapData[, "pvalue"] <- runif(nrow(LocusMapData))

# Define the input list locusMap.list
locusList <- list(snp.var="SNP", chrm.var="CHROMOSOME", loc.var="LOCATION")

# Create the plot
Manhattan.plot(LocusMapData, "pvalue", locusList)
```

---

pheno.list *List to describe the covariate and outcome data*

Description

The list to describe the covariate and outcome data for GxE.scan.

Format

The format is: List of 14

file Covariate data file. This file must have variable names, two of which being an id variable and a response variable (see `id.var` and `response.var`). No default.

id.var Name of the id variable(s). No default.

response.var Name of the binary response variable. This variable must be coded as 0 and 1. No default.

strata.var Stratification variable name or a formula for variables in `file`. See the individual model documentation for the allowable stratifications. The default is NULL so that all observations belong to the same strata.

main.vars Character vector of variables names or a formula for variables in `file` that will be included in the model as main effects. The default is NULL.

int.vars Character vector of variable names or a formula for variables in `file` that will be included in the model as interactions with each SNP in the genotype data. The default is NULL.

file.type 1, 3, 4. 1 is for an R object file created with the `save()` function. 3 is for a table that will be read in with `read.table()`. 4 is for a SAS data set. The default is 3.

delimiter The delimiter in `file`. The default is "".

factor.vars Vector of variable names to convert into factors. The default is NULL.

in.miss Vector of character strings to define the missing values. This option corresponds to the option `na.strings` in `read.table()`. The default is "NA".

subsetData List of sublists to subset the phenotype data for analyses. Each sublist should contain the names "var", "operator" and "value" corresponding to a variable name, operator and values of the variable. Multiple sublists are logically connected by the AND operator. For example,
subsetData=list(list(var="GENDER", operator="==", value="MALE"))
will only include subjects with the string "MALE" for the GENDER variable.
subsetData=list(list(var="AGE", operator=">", value=50),
list(var="STUDY", operator="%in%", value=c("A", "B", "C")))
will include subjects with AGE > 50 AND in STUDY A, B or C. The default is NULL.

cc.var Name of the `cc.var` variable used in `snp.matched`. The default is NULL.

nn.var Name of the `nn.var` variable used in `snp.matched`. The default is NULL.

Details

In this list, `file`, `id.var`, and `response.var` must be specified. The variable `id.var` is the link between the covariate data and the genotype data. For each subject id, there must be the same subject id in the genotype data for that subject to be included in tha analysis. If the genotype data is in a PLINK format, then `id.var` must be of length 2 corresponding the the family id and subject id.

Missing data: If any of the variables defined in `main.vars`, `int.vars`, `strata.var`, or `response.var` contain missing values, then those subjects will be removed from the covariate and outcome data. After the subjects with missing values are removed, the subject ids are matched with the genotype data.

`printEffects` *Print an effects table*

Description

Prints an object returned from `snp.logistic` or `snp.matched`

Usage

```
printEffects(obj, op=NULL)
```

Arguments

`obj`	The return object from `snp.logistic` or `snp.matched`. No default.
`op`	Options list with names "digits" and "method" (see details). The default is NULL.

Details

Below are the names for the options list op. All names have default values if they are not specified.

- digits Integer: Number of significant digits to print. The default is 2.
- method Vector of values from "UML", "CML", "EB" or "CCL", "HCL", "CLR". The default is NULL.

Value

Returns NULL

See Also

snp.effects

Examples

```
# Use the ovarian cancer data
data(Xdata, package="CGEN")

# Fit using a stratification variable
fit <- snp.logistic(Xdata, "case.control", "BRCA.status",
                    main.vars=c("oral.years", "n.children"),
                    int.vars=c("oral.years", "n.children"),
                    strata.var="ethnic.group")

# Compute the effects
effects <- snp.effects(fit, "oral.years", var.levels=c(0, 2, 3))

printEffects(effects)
```

QQ.plot *QQ plot*

Description

Create a quantile-quantile plot

Usage

```
QQ.plot(pvalues, op=NULL)
```

Arguments

pvalues	Vector or matrix of p-values. No default.
op	List of options (see details). The default is NULL.

Details

Plots the ranked p-values against their expected order statistics on a minus log base 10 scale. If `pvalues` is a matrix, then multiple qq plots will be produced, each on a seperate screen. **Options list op:** Below are the names for the options list `op`. All names have default values if they are not specified.

- `cex` Magnification factor for the plotting symbol. The default is 1.
- `cex.axis` Magnification factor for the axes. The default is 1.
- `cex.lab` Magnification factor for the axes labels. The default is 1.
- `cex.main` Magnification factor for the title. The default is 1.
- `color` The color of the plot and title. The default is "blue".
- `figs` Two element vector for the number of rows and columns if `pvalues` is a matrix. The default is NULL.
- `min.p` All p-values less than `min.p` will be set to `min.p`. The default is 1e-16.
- `pch` The plotting symbol. The default is 21 (circle).
- `title` Character string or vector for the title of each plot. If `pvalues` is a matrix, then the column names will be used if `title` is NULL. The default is NULL.
- `xlim` The x-axis limits. The default is determined from `pvalues`.
- `ylim` The y-axis limits. The default is determined from `pvalues`.

Value

Returns 0. Creates a QQ plot.

See Also

`Manhattan.plot`

Examples

```
# set.seed(123)
p <- runif(1000)
QQ.plot(p)
```

---

snp.effects *Joint and Stratified Effects*

Description

Computes joint and stratified effects of the SNP and another variable based on a fitted model.

Usage

```
snp.effects(fit, var, var.levels=c(0, 1), method=NULL)
```

Arguments

fit	Return object from snp.logistic or snp.matched. If fit is the return object from snp.matched, then the snp.vars argument in snp.matched must consist of a single SNP. No default.
var	Name of the second variable to compute the effects for. This variable can be a dummy variable, continuous variable, or a factor. Note that if this variable enters the model as both a main effect and interaction, then it must enter the model the same way as a main effect and interaction for the effects to be computed correctly. For example, if var is a factor as a main effect, then it also must be a factor as an interaction. No default.
var.levels	(For continuous var) Vector of levels. First level is assumed to be the baseline level. The default is c(0, 1).
method	Vector of values from "UML", "CML", "EB" or "CCL", "HCL", "CLR". The default is NULL.

Details

The joint and stratified effects are computed for each method in fit. The stratified effects are the sub-group effect of the SNP stratified by var and the sub-group effect of var stratified by the SNP.

Definition of joint and stratified effects:
Consider the model:

$$logit(P(y = 1)) = \alpha + \beta SNP + \gamma X + \delta SNPX.$$

Let 0 be the baseline for SNP and x_0 the baseline for X. Then the joint effect for SNP = s and X = x relative to SNP = 0 and X = x_0 is

$$\frac{\exp(\alpha + \beta s + \gamma x + \delta sx)}{\exp(\alpha + \gamma x_0)}$$

The stratified effect of the SNP relative to SNP = 0 given X = x is

$$\frac{\exp(\alpha + \beta s + \gamma x + \delta sx)}{\exp(\alpha + \gamma x)}$$

The stratified effect of var relative to X = x given SNP = s is

$$\frac{\exp(\alpha + \beta s + \gamma x + \delta sx)}{\exp(\alpha + \beta s)}$$

A convenient way to print the returned object to view the effects tables is with the function printEffects.

Value

If fit is of class snp.logistic, then the return object is a list of with names "UML", "CML", and "EB". If fit is of class snp.matched, then the return object is a list of with names "CLR", "CCL", and "HCL". Each sublist contains joint effects, stratified effects, standard errors and confidence intervals. The sub-group effect of the SNP stratified by var is in the list "StratEffects", and the sub-group effect of var stratified by the SNP is in the list "StratEffects.2".

See Also

printEffects snp.effects.plot

Examples

```
# Use the ovarian cancer data
data(Xdata, package="CGEN")

# Fit using a stratification variable
fit <- snp.logistic(Xdata, "case.control", "BRCA.status",
                    main.vars=c("oral.years", "n.children"),
                    int.vars=c("oral.years", "n.children"),
                    strata.var="ethnic.group")

# Compute the effects
effects <- snp.effects(fit, "oral.years", var.levels=0:5)
```

---

snp.effects.plot *Effects plot*

Description

Creates a plot of the effects returned from snp.effects

Usage

```
snp.effects.plot(obj.list, op=NULL)
```

Arguments

obj.list	Return object or list of return objects from snp.effects. No default.
op	List of options (see details). The default is NULL.

Details

Plots the effects returned from snp.effects. By default, the effects in StratEffects for each method will be plotted. The side of the effect will have a sawtooth edge if the effect goes beyond the limits of the plot.

Options list op: Below are the names for the options list op. All names have default values if they are not specified.

- method Character vector of the values "UML", "CML", "EB", "HCL", "CCL", "CLR". The default is all methods will be plotted.
- type One of "JointEffects", "StratEffects", "StratEffects.2". The default is StratEffects.
- ylim NULL or a 2-element numeric vector specifying the y-axis limits for all plots. If not specified, different plots will be on different scales. The default is NULL.
- legend See legend. Set to NA for no legend to appear. The default is NULL.

- `split.screen` NULL or a 2-element vector for partitioning the plot window. This option is only valid for inputing a list of objects. The default is NULL.
- `colors` Character vector of colors to use in the plot. See `colors` for all possible colors. The default is NULL.
- `levels1` Vector of levels for the SNP variable to plot. When plotting more than one method, `levels1` has the default value of 1. Otherwise, the default is NULL.
- `levels2` Vector of levels to plot for the variable `var` (in `snp.effects`). The default is NULL.
- `addCI` 0 or 1 to add 95% confidence intervals to the plot. The confidence intervals appear as un-filled boxes around each odds-ratio. The default is 0.

Value

Returns nothing. Creates a snp effect plot.

See Also

`snp.effects`

Examples

```
# Use the ovarian cancer data
data(Xdata, package="CGEN")

# Add some fake SNPs
# set.seed(636)
Xdata[, "rs123"] <- rbinom(nrow(Xdata), 1, 0.4)
Xdata[, "rs456"] <- rbinom(nrow(Xdata), 1, 0.4)
Xdata[, "rs789"] <- rbinom(nrow(Xdata), 1, 0.4)

snpVars <- c("BRCA.status", "rs123", "rs456", "rs789")
objects <- list()
for (i in 1:length(snpVars)) {
  fit <- snp.logistic(Xdata, "case.control", snpVars[i],
                      main.vars=c("oral.years", "n.children"),
                      int.vars=c("oral.years", "n.children"),
                      strata.var="ethnic.group")

  # Compute the effects
  objects[[i]] <- snp.effects(fit, "oral.years", var.levels=0:4)
}

# Plot
snp.effects.plot(objects)

# Plot all on the same scale
#snp.effects.plot(objects, op=list(ylim=c(0.9, 1.4), legend=list(x="bottom")))

# Plot all the joint effects of rs789 for the CML method and add confidence intervals
#snp.effects.plot(objects[[4]], op=list(method="CML", type="JointEffects",
#        legend=list(x="bottomleft", inset=0), ylim=c(0.45, 1.3),
#        colors=c("blue", "aquamarine", "skyblue"), addCI=1))
```

`snp.list` *List to describe the genotype data*

Description

The list to describe the genotype data for `GxE.scan`.

Format

The format is: List of 14

file File to use. No default.

format Genotype format (see details). Typical values are "bed", "tped", "impute", "ldat", "lbat", etc. The default is determined from the `file` extension.

subject.list List to describe the subject ids stored in a file. This list is only needed when the genotype file does not contain the subject ids (for example with PLINK files). The order of the subject ids is `subject.list$file` must match the order in the genotype data. See `subject.list`. The default is NULL.

start.vec Starting row of `file` to begin processing SNPs. The default is 1.

stop.vec Last row of `file` to finish processing SNPs. Use any value < 1 so that all SNPs from rows `start.vec` to the end of the file will be analyzed. The default is -1.

delimiter The delimiter used in `file`. The default is determined from the `file` format.

in.miss Vector of values to denote the missing values in `file`. The default is determined from the `file` format.

heter.codes Vector of codes used for the heterozygous genotype. If NULL, then it is assumed that the heterozygous genotype is of the form "AB", "Aa", "CT", ... etc, ie a 2-character string with different characters (case sensitive). The default is NULL.

Options only used with `GxE.scan`:

include.snps File, list or character vector to define which SNPs should be included in the analysis. If a file, then the file should contain a single column of SNP ids to include. More generally, if the SNPs to be included are in a file with multiple columns, then `include.snps` can be a list of type `subject.list`. If it is a character vector, then it should be a vector of SNP ids. This option can also be used with the options `start.vec` and `stop.vec` (see details). The default is NULL.

PLINK Command for running the PLINK software to transform certain `file` formats (see details). Set PLINK to "" if PLINK is not available or if you do not want PLINK to be used.
The PLINK software can be found at http://pngu.mgh.harvard.edu/~purcell/plink/. The default is "plink".

GLU Command for running the GLU software to transform certain `file` formats (see details). Set GLU to "" if GLU is not available or if you do not want GLU to be used.
The GLU software can be found at http://code.google.com/p/glu-genetics/. The default is "glu".

Details

In this list, `file` must be specified, and `format` should be specified. If not, then the program will attempt to guess the correct `format` of the genotype data from the file extension of `file`. If `format` is a format that `GxE.scan` is not set up to read directly (such as "bed", "lbat", "ped"), then either PLINK or GLU will be called to transform the data into either a "tped" or "ldat" format. When the option `include.snps` is specified as a file, then the options `start.vec` and `stop.vec` will be applied to the SNPs in this file. For example, suppose we have the genotype file snps.bed which is the PLINK "bed" format. We can set `include.snps` to the corresponding ".bim" file:
include.snps <- list(file="snps.bim", id.var=2, header=0, delimiter="\t"). Then the included SNPs in the analysis will be the SNPs in rows `start.vec` to `stop.vec` of file "snps.bim".

Other options such as `delimiter` and `in.miss` do not need to be specified, because they can be determined from the genotype data format. If the SNPs are coded in the standard (0,1,2) coding, then set `heter.codes` to 1 (the heterozygous genotype).

Examples

```
# Example snp.list for a PLINK binary pedigree file when using GxE.scan
## Not run:
pathToPLINK  <- "c:/PLINK/plink-1.07-dos/plink.exe"
snp.file     <- "c:/data/project1/lungCancer.bed"
subject.list <- "c:/data/project1/lungCancer.fam"
snp.list <- list(file=snp.file, format="bed", PLINK=pathToPLINK,
                 subject.list=subject.list)

## End(Not run)

# Suppose the genotype data is an output genotype file from th IMPUTE2 software
# The below list is for processing the file.
## Not run:
snp.list <- list(file="C:/temp/data/chr11_1.imputed.txt.gz", delimiter=" ", format="impute")

## End(Not run)
```

`snp.logistic` *Logistic regression analysis for a single SNP*

Description

Performs logistic regression including a particular SNP (G) and a set of covariates (X) that could include environmental covariates or/and other genetic variables. Included are three analysis options: **(i) Unconstrained maximum-likelihood:** This method is equivalent to prospective logistic regression analysis and corresponds to maximum-likelihood analysis of case-control data allowing the joint distribution of all the factors (the SNP of interest and all other covariates) of the model to be completely unrestricted (non-parametric) **(ii) Constrained maximum-likelihood:** This method performs maximum-likelihood analysis of case-control data under the assumption of HWE and indepenence between the SNP and other factors of the model.The analysis allows the assumptions of HWE and independence to be valid only conditional on certain stratification variables (S), such as

self reported ethnicity or principal compoenets of population stratification. **(iii) Empirical-Bayes:** This method uses an empirical-Bayes type "shrinkage estimation" technique to trade-off bias and variance between the constrained and unconstrained maximum-likelihood estimators.

Usage

```
snp.logistic(data, response.var, snp.var, main.vars=NULL, int.vars=NULL,
             strata.var=NULL, op=NULL, additive.trend=FALSE)
```

Arguments

`data`	Data frame containing all the data. No default.
`response.var`	Name of the binary response variable coded as 0 (controls) and 1 (cases). No default.
`snp.var`	Name of the SNP variable, which must be coded 0-1-2 (or 0-1). The SNP will be included as a main effect in the model. No default.
`main.vars`	Character vector of variable names or a formula for all covariates of interest which need to be included in the model as main effects. The default is NULL, so that only the SNP variable will be included as a main effect in the model.
`int.vars`	Character vector of variable names or a formula for all covariates of interest that will interact with the SNP variable. The default is NULL, so that no interactions will be in the model.
`strata.var`	Name of the stratification variable or a formula (see details for more info). If `strata.var="SVAR"`, where "SVAR" is a factor or character variable in `data`, then "SVAR" will be treated as categorical. Otherwise, "SVAR" is treated as a continuous variable. The default is NULL (1 stratum).
`op`	A list with names `genetic.model`, `reltol`, `maxiter`, and `optimizer` (see details). The default is NULL.
`additive.trend`	Logical, FALSE by default. No need to set. It is used by additive.test() function for additive trend implementation.

Details

Note: Non-dummy continuous variables should be scaled for stability of the algorithm. The `scale` function can be used for this.

The data is first fit using standard logistic regression. The estimated parameters from the standard logistic regression are then used as the initial estimates for the constrained model. For this, the `optim()` function is used to compute the maximum likelihood estimates and the estimated covariance matrix. The empirical Bayes estimates are then computed by combining both sets of estimated parameters (see below). The "strata" option, that is relevent for the CML and EB method, allows the assumption of HWE and G-X independence to be valid only conditional on a given set of other factors. If a single categorical variable name is provided, then the unique levels of the variable will be used to define categorical strata. Otherwise it is assumed that `strata.var` defines a parametric model for variation of allele frequency of the SNP as a function of the variables included. No assumption is made about the relationship between X and S. Typically, S would include self reported ethnicity, study, center/geographic region and principal components of population stratification. The CML method with the "strata" defined by principal compoenents of population stratification

can be viewed as a generalization of adjusted case-only method described in Bhattacharjee et al. (2010). More details of the individual methods follow.

Definition of the likelihood under the gene-environment independence assumption:

Let D = 0, 1 be the case-control status, G = 0, 1, 2 denote the SNP genotype, S denote the stratification variable(s) and X denote the set of all other factors to be included in the regression model. Suppose the risk of the disease (D), given G, X and S can be described by a logistic regression model of the form

$$\log \frac{Pr(D=1)}{Pr(D=0)} = \alpha + Z\beta$$

where Z is the entire design matrix (including G, X, possibly S and their interaction with X) and β is the vector of associated regression coefficients. The CML method assumes Pr(G|X,S)=Pr(G|S), i.e., G and X are conditionally independent given S. The current implementation of the CML method also assume the SNP genotype frequency follows HWE given S=s, although this is not necessary in general. Thus, if f_s denotes the allele frequency given S=s, then

$$P(G=0|S=s) = (1-f_s)^2$$

$$P(G=1|S=s) = 2f_s(1-f_s)$$

$$P(G=2|S=s) = f_s^2.$$

If $\xi_s = \log(f_s/(1-f_s))$, then

$$\log\left(\frac{P(G=1)}{P(G=0)}\right) = \log(2) + \xi_s$$

and

$$\log\left(\frac{P(G=2)}{P(G=0)}\right) = 2\xi_s$$

Chatterjee and Carroll (2005) showed that under the above constraints, the maximum-likelihood estimate for the β coefficients under case-control design can be obtained based on a simple conditional likelihood of the form

$$P^*(D=d, G=g|Z,S) = \frac{\exp(\theta_s(d,g|Z))}{\sum_{d,g}\exp(\theta_s(d,g|Z))}$$

where the sum is taken over the 6 combinations of d and g and $\theta_s(d,g) = d\alpha^* + dZ\beta + I(g=1)\log(2) + g\xi_s$. If S is a single categorical variable, then a separate ξ_s is allowed for each S=s. Otherwise it is assumed $\xi_s = V_s\gamma$, where V_s is the design matrix associated with the stratification and γ is the vector of stratification parameters. If for example, S is specified as "strata=~PC1+PC2+...PCK" where PCk's denote principal components of population stratification, then it is assumed that the allele frequency of the SNP varies in directions of the different principal components in a logistic linear fashion.

Definition of the empirical bayes estimates:
Let β_{UML} be the parameter estimates from standard logistic regression, and let $\eta = (\beta_{CML}, \xi_{CML})$ be the estimates under the gene-environment independence assumption. Let $\psi = \beta_{UML} - \beta_{CML}$, and ϕ^2 be the vector of variances of β_{UML}. Define diagonal matrices of weights to be $W1 =$

$diag(\psi^2/(\psi^2+\phi^2)$ and $W2 = diag(\phi^2/(\psi^2+\phi^2)$, where ψ^2 is the elementwise product of the vector ψ. Now, the empirical bayes parameter estimates are

$$\beta_{EB} = W1\beta_{UML} + W2\beta_{CML}$$

For the estimated covariance matrix, define the diagonal matrix

$$A = diag\left(\frac{\phi^2(\phi^2-\psi^2)}{(\phi^2+\psi^2)^2}\right)$$

where again the exponentiation is the elementwise product of the vectors. If I is the pxp identity matrix and we define the px2p matrix $C = (A, I-A)$, then the estimated covariance matrix is

$$VAR(\beta_{EB}) = C * COV(\beta_{UML}, \beta_{CML}) * C'$$

The covariance term $COV(\beta_{UML}, \beta_{CML})$ is obtained using an influence function method (see Chen YH, Chatterjee N, and Carroll R. for details about the above formulation of the empirical-Bayes method).

Options list:

Below are the names for the options list `op`. All names have default values if they are not specified.

- `genetic.model` 0-3: The genetic model for the SNP. 0=additive, 1=dominant, 2=recessive, 3=general (co-dominant).
- `reltol` Stopping tolerance. The default is 1e-8.
- `maxiter` Maximum number of iterations. The default is 100.
- `optimizer` One of "BFGS", "CG", "L-BFGS-B", "Nelder-Mead", "SANN". The default is "BFGS".

Value

A list containing sublists with names UML (unconstrained maximum likelihood), CML (constrained maximum likelihood), and EB (empirical Bayes). Each sublist contains the parameter estimates (parms) and covariance matrix (cov). The lists UML and CML also contain the log-likelihood (loglike). The list CML also contains the results for the stratum specific allele frequencies under the HWE assumption (strata.parms and strata.cov). The EB sublist contains the joint UML-CML covariance matrix.

References

Mukherjee B, Chatterjee N. Exploiting gene-environment independence in analysis of case-control studies: An empirical Bayes approach to trade-off between bias and efficiency. Biometrics 2008, 64(3):685-94.

Mukherjee B et al. Tests for gene-environment interaction from case-control data: a novel study of type I error, power and designs. Genetic Epidemiology, 2008, 32:615-26.

Chatterjee, N. and Carroll, R. Semiparametric maximum likelihood estimation exploting gene-environment independence in case-control studies. Biometrika, 2005, 92, 2, pp.399-418.

Chen YH, Chatterjee N, Carroll R. Shrinkage estimators for robust and efficient inference in haplotype-based case-control studies. Journal of the American Statistical Association, 2009, 104: 220-233.

Bhattacharjee S, Wang Z, Ciampa J, Kraft P, Chanock S, Yu K, Chatterjee N Using Principal Components of Genetic Variation for Robust and Powerful Detection of Gene-Gene Interactions in Case-Control and Case-Only studies. American Journal of Human Genetics, 2010, 86(3):331-342.

See Also

snp.score, snp.matched

Examples

```
# Use the ovarian cancer data
data(Xdata, package="CGEN")

# Fit using a stratification (categorical) variable
ret <- snp.logistic(Xdata, "case.control", "BRCA.status",
                    main.vars=c("oral.years", "n.children"),
                    int.vars=c("oral.years", "n.children"),
                    strata.var=~factor(ethnic.group))

# Compute a summary table for the models
getSummary(ret)

# Compute a Wald test for the main effect of the SNP and interaction
getWaldTest(ret, c("BRCA.status", "BRCA.status:oral.years", "BRCA.status:n.children"))

# Fit the same model as above using formulas
ret2 <- snp.logistic(Xdata, "case.control", "BRCA.status",
                    main.vars=~oral.years + n.children,
                    int.vars=~oral.years + n.children,
                    strata.var=~factor(ethnic.group))
```

---

snp.matched *Robust G-G and G-E Interaction with Finely-Matched Case-Control Data.*

Description

Performs a conditional likelihood-based analysis of matched case-control data typically modeling a particular SNP and a set of covariates that could include environmental covariates or/and other genetic variables. Three alternative analysis options are included: **(i) Conditional Logistic Regression (CLR):** This method is classical CLR that does not try to utilize G-G or G-E independence allowing the joint distribution of the covariates in the model to be completely unrestricted (non-parametric) **(ii) Constrained Conditional Logistic (CCL) :** This method performs CLR analysis of case-control data under the assumption of gene-environment (or/and gene-gene) independence not in the entire population but within finely matched case-control sets. **(iii) Hybrid Conditional**

Logistic (HCL): This method is suitable if nearest neighbor matching (see the reference by Bhattacharjee et al. 2010) is performed without regard to case-control status. The likelihood (like CCL) assumes G-G/G-E independence within matched sets but in addition borrows some information across matched sets by using a parametric model to account for heterogeneity in disease across strata.

Usage

```
snp.matched(data, response.var, snp.vars, main.vars=NULL, int.vars=NULL,
            cc.var=NULL, nn.var=NULL, op=NULL)
```

Arguments

`data`	Data frame containing all the data. No default.
`response.var`	Name of the binary response variable coded as 0 (controls) and 1 (cases). No default.
`snp.vars`	A vector of variable names or a formula, generally coding a single SNP variable (see details). No default.
`main.vars`	Vector of variable names or a formula for all covariates of interest which need to be included in the model as main effects. The default is NULL, so that only the `snp.vars` will be included as main effect(s) in the model.
`int.vars`	Character vector of variable names or a formula for all covariates of interest that will interact with the SNP variable. The default is NULL, so that no interactions will be in the model.
`cc.var`	Integer matching variable with at most 10 subjects per stratum (e.g. CC matching using `getMatchedSets`) Each stratum has one case matched to one or more controls (or one control matched to one or more cases). The default is NULL.
`nn.var`	Integer matching variable with at most 8 subjects per stratum (e.g. NN matching using `getMatchedSets`) Each stratum can have zero or more cases and controls. But entire data set should have both cases and controls. The default is NULL. At least one of cc.var or nn.var should be provided.
`op`	Control options for Newton-Raphson optimizer. List containing members "maxiter" (default 100) and "reltol" (default 1e-5).

Details

To compute HCL, the data is first fit using standard logistic regression. The estimated parameters from the standard logistic regression are then used as the initial estimates for Newton-Raphson iterations with exact gradient and hessian. Similarly for CCL, the data is first fit using `clogit` using `cc.var` to obtain the CLR estimate as an intial estimate and Newton-Raphson is used to maximize the likelihood.

While `snp.logistic` parametrically models the SNP variable, this function is non-parametric and hence offers somewhat more flexibility. The only constraint on `snp.vars` is that it is independent of `int.vars` within homogenous matched sets. It can be any genetic or non-genetic variable or a collection of those. For example 3 SNPs coded as general, dominant and additive can be specified through a single formula e.g., "snp.vars= ~ (SNP1==1) + (SNP1 == 2) + (SNP2 >= 1)+ SNP3." However, when multiple variables are used in `snp.vars` results should be interpreted carefully.

Summary function snp.effects can only be applied if a single SNP variable is coded.

Note that int.vars consists of variables that interact with the SNP variable and can be assumed to be independent of snp.vars within matched sets. Those interactions for which independence is not assumed can be included in main.vars (as product of appropriate variables).

Both CCL and HCL provide considerable gain in power compared to standard CLR. CCL derives more power by generating pseudo-controls under the assumption of G-G/G-E independence within matched case-control sets. HCL makes the same assumption but allows each matched set to have any number of cases and controls unlike classical case-control matching. By comparing across matched sets, it is able to estimate the intercept parameter and improve efficiency of estimating main effects compared to CLR and CCL. At the same time behaves similar to CCL for interactions by assuming G-G/G-E independence only within mathced sets. For both these methods, the power increase for interaction depends on sizes of the matched sets in nn.var, which is currently limited to 8, to avaoid both memory and speed issues.

The authors would like to acknowledge Bijit Kumar Roy for his help in designing the internal data structure and algorithm for HCL/CCL likelihood computations.

Value

A list containing sublists with names CLR, CCL, and HCL. Each sublist contains the parameter estimates (parms), covariance matrix (cov), and log-likelihood (loglike).

References

Chatterjee N, Zeynep K and Carroll R. Exploiting gene-environment independence in family-based case-control studies: Increased power for detecting associations, interactions and joint-effects. Genetic Epidemiology 2005; 28:138-156.

Bhattacharjee S., Wang Z., Ciampa J., Kraft P., Chanock S, Yu K., Chatterjee N. Using Principal Components of Genetic Variation for Robust and Powerful Detection of Gene-Gene Interactions in Case-Control and Case-Only studies. American Journal of Human Genetics 2010, 86(3):331-342.

Breslow, NE. and Day, NE. Conditional Logistic Regression for Matched Sets. In "Statistical methods in cancer research. Volume I - The analysis of case-control studies." 1980, Lyon: IARC Sci Publ;(32):247-279.

See Also

getMatchedSets, snp.logistic

Examples

```
# Use the ovarian cancer data
data(Xdata, package="CGEN")

# Fake principal component columns
```

```
# set.seed(123)
Ydata <- cbind(Xdata, PC1=rnorm(nrow(Xdata)), PC2=rnorm(nrow(Xdata)))

# Match using PC1 and PC2
mx <- getMatchedSets(Ydata, CC=TRUE, NN=TRUE, ccs.var="case.control",
                     dist.vars=c("PC1","PC2"), size = 4)

# Append columns for CC and NN matching to the data
Zdata <- cbind(Ydata, CCStrat=mx$CC, NNStrat=mx$NN)

# Fit using variable names
ret1 <- snp.matched(Zdata, "case.control",
     snp.vars = "BRCA.status",
                    main.vars=c("oral.years", "n.children"),
                    int.vars=c("oral.years", "n.children"),
                    cc.var="CCStrat", nn.var="NNStrat")


# Compute a Wald test for the main effect of BRCA.status and its interactions

getWaldTest(ret1, c("BRCA.status", "BRCA.status:oral.years", "BRCA.status:n.children"))

# Fit the same model as above using formulas.
ret2 <- snp.matched(Zdata, "case.control", snp.vars = ~ BRCA.status,
                    main.vars=~oral.years + n.children,
                    int.vars=~oral.years + n.children,
                    cc.var="CCStrat",nn.var="NNStrat")

 # Compute a summary table for the models
 getSummary(ret2)
```

`snp.score` *Score Test*

Description

Score tests for genetic association incorporating gene-environment interaction. The function implements two types of score-tests: (1) MScore: A test based on maximum of a class of exposure-weighted score test (Han et al, Biomentrics, 2015) (2) JScore: Joint score-test for genetic association and interaction based on standard logistic model (Song et al., In Prep).

Usage

```
snp.score(data, response.var, snp.var, exposure.var, main.vars,
              strata.var=NULL, op=NULL)
```

Arguments

data	Data frame containing all the data. No default.
response.var	Name of the binary response variable coded as 0 (controls) and 1 (cases). No default.
snp.var	Name of the genotype variable. No default.
exposure.var	Character vector of variable names or a formula for the exposure variables. No default.
main.vars	Character vector of variable names or a formula for all covariates of interest which need to be included in the model as main effects. This argument can be NULL for op$method=1 (see details).
strata.var	Name of the stratification variable for a retrospective likelihood. The option allows the genotype frequency to vary by the discrete level of the stratification variable. Ethnic or geographic origin of subjects, for example, could be used to define strata. Unlike snp.logistic, the strata.var in snp.score cannot currently handle continuous variables like principal components of population stratification.
op	A list of options (see details). The default is NULL.

Details

The MScore option performs a score test for detecting an association between a SNP and disease risk, encompassing a broad range of risk models, including logistic, probit, and additive models for specifying joint effects of genetic and environmental exposures. The test statistics are obtained my maximizing over a class of score tests, each of which involves modified standard tests of genetic association through a weight function. This weight function reflects the potential heterogeneity of the genetic effects by levels of environmental exposures under a particular model. The MScore test could be performed using either a retrospective or prospective likelihood depending on whether an assumption of gene-environment independence is imposed or not, respectively. The JScore function performs joint score-test for genetic association and gene-environment interaction under a standard logistic regression model. The JScore test could be performed under a prospective likelihood that allows association between gene and environment to remain unrestricted, a retrospective likelihood that assumes gene-environment independence and an empirical-Bayes framework that allows data adaptive shrinkage between retrospective and prospective score-tests. The JScore function is explicitly developed for proper analysis of imputed SNPs under all of the different options. The MScore function should produce valid tests for imputed SNPs under prospective likelihood. But further studies are needed for this method for analysis of imputed SNPs with the retrospective likelihood. The JScore function returns one-step MLE for parameters which can be used to perform meta-analysis across studies using standard techniques.

Options list:

Below are the names for the options list op.

- method 1 or 2 for the test. 1 = MScore, 2 = JScore. The default is 2.

Options for method 1:

- `thetas` Numeric vector of values in which the test statistic will be calculated over to find the maximum. Theta values correspond to different risk models, which can take any value of real numbers. For example, theta = -1 corresponds to an additive model, 0 to a multiplicative model, and a probit model is between -1 < theta < 0. Supra-multiplicative model corresponds to theta > 1. The default is seq(-3, 3, 0.1)
- `indep` TRUE or FALSE for the gene-environment independence assumption. The default is FALSE.
- `doGLM` TRUE or FALSE for calculating the Wald p-value for the SNP main effect. The default is FALSE.

Options for method 2:

- `sandwich` TRUE or FALSE to return tests and p-values based on a sandwich covariance matrix. The default is FALSE.

Value

For `op$method` = 1 (MScore), the returned object is a list with the following components:

- `maxTheta` Value of `thetas` where the maximum score test occurs.
- `maxScore` Maximum value of the score test.
- `pval` P-value of the score test.
- `pval.logit` P-value of the standard association test based on logistic regression.
- `model.info` List of information from the model.

For `op$method` = 2 (JScore), the returned object is a list containing test statistics, p-values and one-step MLEs for the parameters and variance-covariance matrices for the UML, CML and EB methods. Any name in the list containing the string "test", "pval", "parm" or "var" is a test statistic, p-value, parameter estimate or variance-covariance matrix respectively.

References

Han, S.S., Rosenberg, P., Ghosh, A., Landi M.T., Caporaso N. and Chatterjee, N. An exposure weighted score test for genetic association integrating environmental risk-factors. Biometrics 2015 (Article first published online: 1 JUL 2015 | DOI: 10.1111/biom.12328)

Song M., Wheeler B., Chatterjee, N. Using imputed genotype data in joint score tests for genetic association and gene-environment interactions in case-control studies (In preparation).

See Also

`snp.logistic`

Examples

```
# Use the ovarian cancer data
data(Xdata, package="CGEN")

table(Xdata[, "gynSurgery.history"])

# Recode the exposure variable so that it is 0-1
temp <- Xdata[, "gynSurgery.history"] == 2
Xdata[temp, "gynSurgery.history"] <- 1

out <- snp.score(Xdata, "case.control", "BRCA.status", "gynSurgery.history",
              main.vars=c("n.children","oral.years"), op=list(method=2))
```

`subject.list` *List to describe the file of subject ids*

Description

The list to describe the file of subject ids for `snp.list`

Format

The format is: List of 4

file Text file containing the subject ids. The file can be a single column of ids, or a delimited file of several columns with the ids as one of the columns. No default.

id.var Column number(s) or variable name(s) containing the subject ids. Use `id.var=-1` if the file is a single column of ids.

delimiter The delimiter in `file`. The default is "".

header 0 or 1 if the file contains a header of variable names.

Details

This list is should only be used when the genotype file does not contain subject ids. The order of the ids in this file must match the order of the genotypes in the genotype file. If the genotype data is in a PLINK format, then `id.var` must be of length 2 corresponding the the family id and subject id. When using the `GxE.scan` function, this list can often just be set to the name of the file containing the ids.

Xdata *Sample covariate and outcome data*

Description

Sample covariate and outcome data

Details

The data is taken from an ovarian cancer study. The file Xdata.txt is a tab-delimited type 3 data set (see `file.type` in `pheno.list`). It contains the variables:

- `id` The subject id
- `case.control` Ovarian cancer status (0, 1)
- `BRCA.status` Simulated data for breast cancer status (0, 1)
- `oral.years` Years of oral contraceptive use
- `n.children` Number of children
- `age.group` Age group in 5 categories (1-5)
- `ethnic.group` Ethnic group in 3 categories (1-3)
- `BRCA.history` Personal history of breast cancer (0,1)
- `gynSurgery.history` History of gynechological surgery (0, 1, 2)
- `family.history` Family history of breast/ovarian cancer (0, 1, 2)

Examples

```
# Load and print the first 5 rows
data(Xdata, package="CGEN")

Xdata[1:5, ]
```

Xdata2 *Sample covariate and outcome data*

Description

Sample covariate and outcome data

Details

The data is taken from a lung cancer study. The file Xdata2.txt is a tab-delimited type 3 data set (see `file.type` in `pheno.list`). It contains the variables:

- `case.control` Lung cancer status (0, 1)
- `SNP` Simulated SNP data
- `smoking` Smoking status, 0: never smokers 1: ever smokers
- `study` Different studies from which subjects are taken: 1, 2, 3, 4, 5
- `cov1` A covariate taking integer values
- `cov2` A covariate taking binary values 0, 1
- `cov3` A covariate taking integer values
- `cov4` A covariate taking continuous values

Examples

```
# Load and print the first 5 rows
data(Xdata2, package="CGEN")

Xdata2[1:5, ]
```

Index

∗ **data**
  LocusMapData, 20
  Xdata, 41
  Xdata2, 41
∗ **distance**
  getMatchedSets, 10
∗ **misc**
  getSummary, 12
  getWaldTest, 13
  GxE.scan.combine, 16
  locusMap.list, 19
  Manhattan.plot, 21
  pheno.list, 22
  printEffects, 23
  QQ.plot, 24
  snp.effects, 25
  snp.effects.plot, 27
  snp.list, 29
  subject.list, 40
∗ **models**
  additive.test, 2
  GxE.scan, 14
  GxE.scan.partition, 17
  snp.logistic, 30
  snp.matched, 34
  snp.score, 37
∗ **package**
  CGEN, 7

additive.test, 2, *8*, *15*

CGEN, 7
clogit, *35*
colors, *21*, *28*

daisy, *10, 11*
dist, *10, 11*

getMatchedSets, *8, 9*, 10, *35, 36*
getSummary, *8, 9*, 12
getWaldTest, *8, 9*, 13
getwd, *18*
GxE.scan, *8*, 14, *17–19*, *21, 22*, *29, 30*, *40*
GxE.scan.combine, *8*, *16*, 16, *19*
GxE.scan.partition, *8*, *16, 17*, 17

hclust, *11*

legend, *27*
locusMap.list, *9*, 19, *21, 22*
LocusMapData, *8*, 20

Manhattan.plot, *8, 9*, *19, 20*, 21, *25*

pheno.list, *8*, *14*, *17, 18*, 22, *41, 42*
points, *21*
print, *9*
printEffects, *9*, 23, *26, 27*

QQ.plot, *8*, *22*, 24

scale, *31*
snp.effects, *8, 9*, *24*, 25, *27, 28*, *36*
snp.effects.plot, *8*, *27*, 27
snp.list, *8*, *14, 15*, *17, 18*, 29, *40*
snp.logistic, *5*, *8*, *12, 13*, *15*, *23*, *26*, 30, *35, 36*, *38, 39*
snp.matched, *8, 9*, *11–13*, *15*, *23*, *26*, *34*, 34
snp.score, *5*, *7, 8*, *14, 15*, *34*, 37
subject.list, *8*, *29*, 40
summary, *9*

Xdata, *8*, 41
Xdata2, 41