CGEN

October 25, 2011

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

Description

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. 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 `snp.logistic` and `snp.scan.logistic`. Whereas `snp.logistic` analyzes one SNP with each function call, `snp.scan.logistic` analyzes a collection of SNPs and writes the summary results to an external file. With `snp.logistic`, a data frame is input in which the SNP variable must be coded as 0-1-2 (or 0-1). If not, `recode.geno` can be used for recoding the SNP variable before calling `snp.logistic`. 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 `snp.scan.logistic`, the data is read in from external files defined in `snp.list` and `pheno.list`. The collection of p-values computed in `snp.scan.logistic`, can be plotted using the functions `QQ.plot` and `chromosome.plot`.
The function for analysis of matched case-control data is `snp.matched`. Optimal matching can be obtained from the function `getMatchedSets`. This package contains sample genotype data `SNPdata`, sample covariate data `Xdata`, and sample SNP meta data `LocusMapData`. The current version of the package is only suitable for analysis of SNPs on non-sex chromosomes.

Author(s)

Samsiddhi Bhattacharjee, Nilanjan Chatterjee and William Wheeler <wheelerb@imsweb.com>

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.

`LocusMapData` *Locus map data*

Description

Locus map data for `chromosome.plot`

Details

LocusMapData.txt is a tab delimited file that contains the chromosome and location information for each SNP in `SNPdata`. 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, ]
```

`QQ.plot` *QQ plot*

Description

Create a quantile-quantile plot

Usage

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

Arguments

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

- `title` Character string for the title of the plot. The default is "QQ PLOT".
- `color` The color of the plot. The default is "blue".

See Also

`chromosome.plot`

Examples

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

SNPdata *Sample genotype data*

Description

Sample genotype data for `snp.scan.logistic`

Details

GenotypeData.txt is a type 2 data file (see `file.type` in `snp.list`). This data contains 230 SNPs and 1579 subjects, and is delimited by a bar ("|"). The first row of the data contains the subject ids. Starting from row 2, are the SNP ids and the genotypes for each subject. The genotypes are coded as 0, 1, or 2 with NA to denote a missing genotype. The data for the first 3 SNPs and 5 subjects are:

sub455|sub1244|sub645|sub1392|sub1482
rs11102647|NA|AT|AT|TT|AA
rs6695241|CC|CG|CC|CC|CG
rs12567796|NA|NA|CG|NA|CG

Examples

```
# Load and print a substring the first 5 lines
data(SNPdata, package="CGEN")

substring(SNPdata[1:5], 1, 50)
```

Xdata *Sample covariate and outcome data*

Description

Sample covariate and outcome data for `snp.scan.logistic`

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, ]
```

`chromosome.plot` *Chromosome plot*

Description

Creates a chromosome plot

Usage

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

Arguments

`infile`	Output file from `snp.scan.logistic`. 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.

- `splitScreen` 0 or 1 to split the plot into two seperate parts. The default is 1.
- `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.
- `subset` Vector of chromosomes to plot. The default is NULL.
- `colors` Character vector of colors to use in the plot. See `colors` for all possible colors. The default is NULL.
- `pch` Vector of plotting symbols to use. See `points` for the different plotting symbols. The default is that circles (pch = 21) will be plotted.

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
chromosome.plot(LocusMapData, "pvalue", locusList)
```

`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"
size = size, ratio = ratio, fixed=TRUE)
  mx$NN[1:10]
  mx$tblNN

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

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

Description

The list to describe the locus map data for `chromosome.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.

sas.list See `sas.list`. The default is NULL.

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 flat file.
- `Type 4` A SAS data set (see `sas.list`).

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

Description

The list to describe the covariate and outcome data for `snp.scan.logistic`

Format

The format is: List of 11

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. 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 `strata.var` argument in `snp.logistic` for more details. 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".

sas.list See `sas.list`.

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.
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)
```

`recode.geno` *Recode a vector of genotypes*

Description

Recodes a vector of genotypes for a single SNP

Usage

```
recode.geno(vec, in.miss=c("  "), out.miss=NA, out.genotypes=c(0,1,2),
                        heter.codes=NULL, subset=NULL)
```

Arguments

`vec`	Vector of genotypes. No default
`in.miss`	Vector of categories which denote missing values in `vec`. These categories will be changed to `out.miss` provided `out.miss` is not NULL. The default is " " (2 spaces).
`out.miss`	New category for the missing values. Use NULL to keep the original missing values. The default is NA.
`out.genotypes`	Vector of length 3 containing the new genotypes. The first element is for the major homozygous genotype, the second element is for the heterozygous genotype, and the third is for the minor homozygous genotype. Use NULL for no recoding of the genotypes; only the missing values will be changed. The default is c(0, 1, 2).
`heter.codes`	Vector of codes in `vec` used for the heterozygous genotype. If NULL, then it is assumed that the heterozygous genotype is of the form "AB", "Aa", "CT", etc (a 2-character string with different characters). The default is NULL.
`subset`	Logical vector of length length(vec) to be used in determining the major and minor homozygous genotypes. This option is useful for case-control studies where typically only the controls are used to determine the major and minor alleles. The default is NULL so that all (non-missing) elements of vec will be used.

Details

The input vector `vec` can be either character or numeric. If it is numeric, then `heter.codes` should be set. This function is useful if the input vector of genotypes is of the form (AA, AB, BB) and the standard (0, 1, 2) coding is desired, or vice-versa.

Value

Vector of recoded genotypes.

Examples

```
# CC is the major homozygous genotype
vec <- c("CC", "TT", "CC", "CT", "CT", "CC", "  ", "TT", "CT", "CC")
vec2 <- as.integer(recode.geno(vec)$vec)
print(vec2)

# Get vec from vec2
recode.geno(vec2, in.miss=NA, out.miss="  ", out.genotypes=c("CC", "CT", "TT"),
            heter.codes=1)$vec

vec <- c("CC", "TT", "!!", "CT", "CT", "CC", "NA", "TT", "CT", "CC")
recode.geno(vec, in.miss=c("!!", "NA"))$vec

vec <- c(0, 2, -9, 1, 1, 0, -9, 2, 1, 0)
recode.geno(vec, in.miss=-9, heter.codes=1)$vec

vec <- c("C/C", "T/T", "C/C", "C/T", "C/T", "C/C", " / ", "T/T", "C/T", "C/C")
recode.geno(vec, in.miss=" / ", heter.codes="C/T")$vec
```

`sas.list` *List for SAS data sets*

Description

The list to use for type 4 (SAS) data sets

Format

The format is: List of 3

sas.exe The complete path to the executable file to start SAS. If there is a command (eg sas) to start SAS from any directory, then use the command instead. No default.

sas.file Path to the file sasfile.sas. This file can be found in the exec subfolder of the CaseControl.Genetics library. No default.

shell The default is "bash".

Details

In this list, `sas.exe` and `sas.file` must be specified. The option `shell` is only useful when running on UNIX.

`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 = x0 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`

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.list` *List to describe the genotype data*

Description

The list to describe the genotype data for `snp.scan.logistic`

Format

The format is: List of 7

file File to use. No default.

file.type 1-8 (see details). The default is 2.

delimiter The delimiter used in `file`. The default is "\t" (a tab).

in.miss Vector of values to denote the missing values in `file`. The default is " " (2 blank spaces).

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.

id.var (Only for `file.type` = 3, 4, 6, 8) The subject id variable. No default.

sas.list See `sas.list`. The default is NULL.

Details

In this list, `file` must be specified. If the SNPs are coded in the standard (0,1,2) coding, then set `heter.codes` to 1 (the heterozygous genotype).

`Types 1, 2, 5, 7` have data in the form:

	subject1	subject2	subject3
snp1	0	2	1
snp2	1	1	0

The first row must contain the subject ids. Starting from row 2, the first delimited field must contain the SNP id. The remaining delimited fields contain the genotypes. Rows are SNPs, columns are the subjects.

- `Type 1` An .rda file created with the `save()` command.
- `Type 2` A flat file that is in the form of the example above.
- `Type 5` A file compressed with WinZip.
- `Type 7` A file compressed with gzip.

`Types 3, 4, 6, 8` have data of the form:

id	snp1	snp2
subject1	0	1
subject2	2	1
subject3	1	0

There must also be a column containing the subject ids (see `id.var`) for these types.

- `Type 3` A flat file that is in the form of the example above.
- `Type 4` A SAS data set (see `sas.list`).
- `Type 6` A file compressed with WinZip.
- `Type 8` A file compressed with gzip.

Examples

```
# Suppose the genotype data is a tab-delimited, type 2 file: c:/temp/data/geno1.txt.
#  Also assume the data has the trend coding 0, 1, 2 with NA as missing values.
# The below list is for processing the file.
## Not run:
snp.list <- list(file="C:/temp/data/geno1.txt", delimiter="\t", file.type=2,
                 heter.codes=1, in.miss=NA)

## 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)
```

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). This variable 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). The default is NULL (1 stratum).
`op`	A list with names `genetic.model`, `reltol`, `maxiter`, and `optimizer` (see details). The default is NULL.

Details

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 variable name is provided, then the unique level of the variable will be used to define categorical strata. If a formula object is given, then it is assumed that the formula describes a parametric model for variation of allele frequency of the SNP as a function of the variables included in the formula. No assumption is made about the relationship between X and factor in 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 compoeenents 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. If S is specified using a formula object, then it is assumed $\xi_s = V_s\gamma$, where V_s is the design matrix associated with the formula object 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=trend, 1=dominant, 2=recessive, 3=general.
- `reltol` Stopping tolerance. The default is 1e-6.
- `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 (log-like). The list CML also contains the results for the stratum specific allele frequencies under the HWE assumption (strata.parms and strata.cov).

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.scan.logistic`, `snp.matched`

Examples

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

# Fit using a stratification 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="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="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 "max-iter" (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",

# 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.scan.logistic *Logistic regression analysis for an array of SNPs*

Description

Performs a logistic regression analysis of case-control data with three alternative 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 the covariates in 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 gene-environment (or/and gene-gene) independence and Hardy-Weinberg-Equilibrium for the underlying population. The analysis allows the assumptions to be valid conditional on a stratification variable **(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.scan.logistic(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

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 genotype data is read in from the file(s) `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 model are defined in `pheno.list`. For an included subject with id `sub.id`, there must be the same id in the genotype data file(s). The genotype data file(s) can contain more subject ids than in `pheno.list$file`, and the ids do not have to be in any particular order. Once the data is read in, all missing values are removed and the function `snp.logistic` is called for each SNP in the genotype data file(s). By default, output files are not created and only the analysis from the last SNP is returned from this function; so to save the results for all the SNPs, the user must specify `op$out.file` or `op$out.dir`.

Options list op: 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=trend, 1=dominant, 2=recessive, 3=general.
- `tests` List of character vectors that will be used in Wald tests. For example, `tests`=list(c("x1", "x2"), c("x1", "x4", "x9")), will compute a 2 df Wald test involving the variables x1 and x2, and will compute a 3 df Wald test for the variables x1, x4, and x9. The variable name for the main effect of each SNP is called "SNP_", and the variable names that interact with each SNP are of the form "SNP_x1", "SNP_gender", etc. In the output, these tests will labeled as "test1", "test2", etc. The default is NULL.
- `tests.1df` Character vector of variable names to compute 1 degree of freedom Wald tests for. The default is NULL.
- `effects` List for joint/stratified effects. The default is NULL. Names in the list must be:
  - `var` Variable name to compute the effects with the SNP variable. This variable must be a main effect. No default.
  - `type` 1, 2 or c(1, 2), 1 = joint, 2 = stratified. The default is 1.
  - `var.levels` (Only for continuous `var`). Numeric vector of the levels to be used in the calculation. The default is 0.
  - `var.base` (Only for continuous `var`). Baseline level. The default is 0.

  - `snp.levels` A vector containing any of the values 0, 1, 2 to use as the levels of each SNP. The default is 1.
  - `method` Character vector containing any of the following: "UML", "CML", "EB". The default is c("UML", "CML", "EB").

- `out.file` NULL or file name to save summary information for each SNP. The output will at least contain the columns "SNP" and "MAF". MAF is the minor allele frequency from the controls. Additional columns in this file are based on the values of `tests` and `tests.1df`. The default is NULL.
- `out.dir` NULL or the output directory to store the output lists for each SNP. A seperate file will be created for each SNP in the SNP data set, so this option should only be used for analyzing a small number of SNPs. The file names will be out_<SNP>.rda. The `load()` function must be used to read these files into R. The object names are called "ret". The default is NULL.
- `reltol` Stopping tolerance. The default is 1e-6.
- `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 from the LAST analysis performed. This list will contain the estimated parameters, covariance matrices, SNP name, and possibly the results of any Wald tests.

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.logistic`

Examples

```
# Define the list for the genotype data.
snp.list <- list()
snp.list$file <- system.file("sampleData", "SNPdata.rda", package="CGEN")
snp.list$file.type <- 1
snp.list$delimiter <- "|"
snp.list$in.miss <- "NA"

# 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()

# Omnibus Wald test for the main effect of the SNP and the interaction variables, and
#  a seperate Wald test for "age.group" and "oral.years".
op$tests <- list(c("SNP_", "SNP_:n.children"), c("age.group", "oral.years"))

# Specifying out.dir will create a separate .rda file for each SNP
#op$out.dir <- "./"
# Specifying out.file will create one output file
#op$out.file <- "out.txt"

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

Index