Package 'ASSET'

April 14, 2017

Title An R package for subset-based association analysis of heterogeneous traits and subtypes

Version 1.12.0

Date 2015-06-01

Author Samsiddhi Bhattacharjee, Nilanjan Chatterjee and William Wheeler

Description An R package for subset-based analysis of heterogeneous traits and subtypes.

Maintainer William Wheeler <WheelerB@imsweb.com>

Depends MASS, msm, rmeta, mvtnorm, tmvnsim

Suggests RUnit, BiocGenerics

License GPL-2 + file LICENSE

biocViews Software, Bioinformatics

NeedsCompilation no

R topics documented:

`ASSET` *Association analysis for SubSETs*

Description

ASSET is a suite of statistical tools specifically designed to be powerful for pooling association signals across multiple studies when true effects may exist only in a subset of the studies and possibly in opposite directions across studies. The method explores all possible subsets (or a restricted set if the user specifies so) of studies and evaluates fixed-effect meta-analysis-type test-statistics for each subset. The final test-statistic is obtained by maximizing the subset-specific test-statistics over all possible subsets and then evaluating its significance after efficient adjustment for multiple-testing, taking into account the correlation between test-statistics across different subsets due to overlapping subjects. The method not only returns a p-value for significance for the overall evidence of association of a SNP across studies, but also outputs the "best subset" containing the studies that contributed to the overall association signal. For detection of association signals with effects in opposite directions, ASSET allows subset search separately for positively- and negatively- associated studies and then combines association signals from two directions using a chi-square test-statistic. The method can take into account correlation due to overlapping subjects across studies (e.g. shared controls). Although the method is originally developed for conducting genetic association scans, it can also be applied for analysis of non-genetic risk factors. Since version 2.0.0, a new option has been added to enable pre-screening of studies based on marginal association tests followed by subset-based meta-analysis to achieve an increase computational speed while analyzing a large number of studies.

Details

The new version includes option to pre-screen phenotypes based on significance of marginal association tests. It increases computational speed for ASSET when analyzing larger number of phenotypes.

The package consists of two main functions: (1) `h.traits` and (2) `h.types`. The function `h.traits` is suitable for conducting meta-analysis of studies of possibly different traits when summary level data are available from individual studies. The function allows correlation among different studies/traits, which, for example, may arise due to shared subjects across studies. The function can also be used to conduct "meta-analysis" across multiple correlated traits on the same individuals by appropriately specifying the correlation matrix for the multivariate trait. The method, however, is not optimized yet (from a power perspective) for analyzing multivariate traits measured on the same individuals. The function `h.types` is suitable for analysis of case-control studies when cases consist of distinct disease subtypes. This function assumes individual level data are available. The functions `h.summary` and `h.forestPlot` are useful for summarizing results and displaying forest plots. The helper functions `z.max` and `p.dlm` are generic functions called internally for obtaining the maximized subset-based test-statistics and the corresponding p-values approximated by the Discrete Local Maximization (DLM) method. These functions can be further customized for specific applications. For example, the default options of these functions currently assume all possible subsets are to be searched. For analysis of case-control studies with ordered diseased subtypes (e.g. stages of a cancer), however, it may be more meaningful to restrict the subset search to incorporate ordering constraints among the disease subtypes. In such a situation, one can pass a function argument `sub.def` to `z.max` and `p.dlm` for performing restricted subset searches.

Author(s)

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

References

Samsiddhi Bhattacharjee, Preetha Rajaraman, Kevin B. Jacobs, William A. Wheeler, Beatrice S. Melin, Patricia Hartge, GliomaScan Consortium, Meredith Yeager, Charles C. Chung, Stephen J.

Chanock, Nilanjan Chatterjee. A subset-based approach improves power and interpretation for combined-analysis of genetic association studies of heterogeneous traits. Am J Hum Genet, 2012, 90(5):821-35

Bhattacharjee et al. Pre-screening and Meta-analysis based on SubSETs: A Fast and Powerful Approach to Pleiotropic Analysis Across a Large Number of Traits (In Preparation)

ex_trait *Data for the h.traits example*

Description

Data for h.traits

Details

The object data contains estimated log odds-ratios and their standard errors for 5 SNPs and 6 traits. The matrices N00, N10, and N11 are the case-control overlap matrices.

See Also

h.traits

Examples

```
data(ex_trait, package="ASSET")

# Display the data, and case/control overlap matrices
data
N00
N11
N10
```

ex_types *Data for the h.types example*

Description

Sample data for h.types

Details

The data frame contains columns for disease subtype, study center, and 3 SNPs.

See Also

h.traits

Examples

```
data(ex_types, package="ASSET")

# Display the first 10 rows of the data
data[1:10, ]
```

`h.forestPlot` *Forest plot for meta-analysis of heterogenerous traits or types.*

Description

Forest Plot for meta-analysis of heterogenerous traits or types.

Usage

```
h.forestPlot(rlist, snp.var, level=0.05, p.adj=TRUE, digits=2)
```

Arguments

`rlist`	The list of results returned by `h.traits` or `h.types`. SNPs other than `snp.var` are ignored.
`snp.var`	A character string giving the name of the SNP variable to be plotted. No default.
`level`	Level for confidence intervals. Default is `0.05` for `95%` confidence intervals.
`p.adj`	Logical. Whether to report Bonferroni adjusted p-values for each individual subtype. Default is TRUE.
`digits`	Number of significant digits to display the odds ratios in the plot.

Value

Forest plot for a SNP showing regression coefficients (e.g. log-odds-ratio for case-control studies) for individual studies/traits and confidence intervals, estimate of an overall regression coefficient and confidence interval based on standard fixed-effect meta-analysis and estimate of regression coefficient(s) and confidence intervals associated with the identified best subset(s).

See Also

`h.summary`, `h.traits`, `h.types`

Examples

```
# Use the example data
data(ex_trait, package="ASSET")
data

# Define the input arguments to h.traits
snps       <- as.vector(data[, "SNP"])
traits.lab <- paste("Trait_", 1:6, sep="")
beta.hat   <- as.matrix(data[, paste(traits.lab, ".Beta", sep="")])
sigma.hat  <- as.matrix(data[, paste(traits.lab, ".SE", sep="")])
```

```
cor        <- list(N11=N11, N00=N00, N10=N10)
ncase      <- diag(N11)
ncntl      <- diag(N00)

# Now let us call h.traits on these summary data.
res <- h.traits(snps, traits.lab, beta.hat, sigma.hat, ncase=ncase,
                ncntl=ncntl, cor=cor, cor.numr=FALSE, search=NULL,
                side=2, meta=TRUE, zmax.args=NULL)

h.forestPlot(res, "SNP_1", digits=3)
```

h.summary *Summary results from subset-search.*

Description

This function produces summary results from subset-based association analysis.

Usage

```
h.summary(rlist, level = 0.05, digits = 3)
```

Arguments

rlist	List returned by h.traits or h.types
level	Level for confidence intervals. Default is 0.05 for 95% confidence intervals. The confidence intervals are obtained by inverting the corresponding multiple-testing adjusted p-values.
digits	Number of significant digits to retain in odds ratios and confidence intervals in the summary table

Details

Returns a list of data frames containing p-values, odds-ratios, confidence intervals and the traits/types for each analysis. The number of data frames in the list will depend on which function (h.traits or h.types) was called and on the function options specified.

Value

A list of data frames, one for each of the methods specified the original call of the functions h.traits or h.types. Each row of a data frame corresponds to a SNP and the values include p-values for overall association (including component-wise p-values for two-sided search), names of phenotypes or disease subtypes included in the best-subset, summary regression coefficients (e.g. log-odds-ratio for case-control studies) representing strength of association of a SNP with the identified subset of traits/subtype and corresponding confidence intervals.

See Also

h.forestPlot, h.traits, h.types

Examples

```
# Use the example data
data(ex_trait, package="ASSET")

# Define the input arguments to h.traits
snps       <- as.vector(data[, "SNP"])
traits.lab <- paste("Trait_", 1:6, sep="")
beta.hat   <- as.matrix(data[, paste(traits.lab, ".Beta", sep="")])
sigma.hat  <- as.matrix(data[, paste(traits.lab, ".SE", sep="")])
cor        <- list(N11=N11, N00=N00, N10=N10)
ncase      <- diag(N11)
ncntl      <- diag(N00)

# Now let us call h.traits on these summary data.
res <- h.traits(snps, traits.lab, beta.hat, sigma.hat, ncase=ncase,
                ncntl=ncntl, cor=cor, cor.numr=FALSE, search=NULL,
                side=2, meta=TRUE, zmax.args=NULL)

h.summary(res)
```

---

h.traits *Heterogeneous traits or studies*

Description

Performs one-sided or two-sided subset-based meta-analysis of heteregeneous studies/traits.

Usage

```
h.traits(snp.vars, traits.lab, beta.hat, sigma.hat, ncase=NULL, ncntl=NULL,
  cor=NULL, cor.numr=FALSE, search=NULL, side=2, meta=FALSE,
  zmax.args=NULL, pval.args=NULL, p.bound=1,
  NSAMP=5000, NSAMP0=50000)
```

Arguments

`snp.vars` A character vector giving the SNP names to be analyzed. No default.

`traits.lab` A character vector giving the names/identifiers of the k studies/traits being analyzed. The order of this vector must match the columns of `beta.hat` and `sigma.hat` No default.

`beta.hat` A matrix of dimension length(`snp.vars`) by (k) (or a vector of length k when 1 SNP is passed). Each row gives the coefficients obtained from the analysis of that SNP across the k studies/traits. No default.

`sigma.hat` A vector or matrix of same dimension as beta.hat, giving the corresponding standard errors. No default.

`ncase` The number of cases in each of the k studies. This can be same for each SNP, in which case `ncase` is a vector of length k. Alternatively if the number of non-missing cases analyzed for each SNP is known, then `ncase` can be a length(`snp.vars`) by (k) matrix. If NULL, then `ncase` will be set to `1/sigma.hat^2`. The default is NULL.

`ncntl` Same as `ncase` (above) for controls. The default is NULL.

`cor` Either a `k` by `k` matrix of inter-study phenotypic correlations or a list containing three case/control overlap matrices (for case-control studies) named `N11`, `N00` and `N10`. See details. Default is NULL, so that studies are assumed to be independent. The rows and columns of all matrices needs to be in the same order as `traits.lab` and columns of `beta.hat` and `sigma.hat`

`cor.numr` Logical. When specified as TRUE the correlation information is used for optimal weighting of the studies in the definition of the meta-analysis test-statistics. The default is FALSE. In either case, the correlation is accounted for variance calculation of the meta-analysis test-statistic.

`search` 1, 2 or NULL. Search option 1 and 2 indicate one-sided and two-sided subset-searches respectively. The default option is NULL that automatically returns both one-sided and two-sided subset searches. Default is NULL.

`side` Either 1 or 2. For two-tailed tests (where absolute values of Z-scores are maximized), `side` should be 2. For one-tailed tests, `side` should be 1 (positive tail is assumed). Default is 2. The option is ignored when `search=2` since the two-sided subset search is automatically a two-tailed test.

`meta` Logical. When specified as TRUE, standard fixed effect meta-analysis results are returned together with results from subset-based meta-analysis. The Default is FALSE.

`zmax.args` Optional arguments to be passed to `z.max` as a named list. This option can be useful if the user wants to restrict subset searches in some structured way, for example, incorporating some ordering constraints.

`pval.args` Optional arguments to be passed to `p.dlm` as a named list. This option can be useful if the user wants to restrict subset searches in some structured way, for example, incorporating some ordering constraints.

`p.bound` P-value threshold for screening studies based on marginal association before performing subset search. Default is 1, that is all studies are included in the subset search. The p-value for the overall procedure accounts for this pre-screening step. See details.

`NSAMP` Number of samples from a truncated multivariate normal distribution used to compute the DLM p-value. The default is 5000.

`NSAMP0` Number of samples from truncated multivariate normal distribution used to calculate the probability of the truncation region in DLM p-value calculation. For 1-sided subset search this is ignored unless `p.bound` < 1. The default is 50000. See details.

Details

The one-sided subset search maximizes the standard fixed-effect meta-analysis test-statistics over all possible subsets (or over a restricted set of subsets if such an option is specified) of studies to detect the best possible association signals. The p-value returned for the maximum test-statistics automatically accounts for multiple testing penalty due to subset search and can be taken as an evidence of an overall association for the SNP accross the `k` studies/traits. The one-sided method automatically guarantees identification of studies/traits that have associations in the same direction and thus is useful in applications where it is desirable to identify SNPs that shows effects in the same direction across multiple traits/studies. The two-sided subset search, applies one-side subset search separately for positively and negatively associated traits for a given SNP and then combines the association signals from two directions into a single combined chi-square type statistic. The method is sensitive in detecting SNPs that may be associated with different traits in different directions.

The methods allow for accounting for correlation among studies/subject that might arise due to shared subjects across distinct studies or due to correlation among related traits in the same study. For application of the method for meta-analysis of case-control studies, the matrices `N11`, `N10` and `N00` denote the number subjects that are shared between studies by case-control status. By defintion, the diagonals of the matrices `N11` and `N00` contain the number of cases and controls, respectively, in the `k` studies. Also, by definition, the diagonal of `N10` is zero since cases cannot serve as controls and vice versa in the same study. The most common situation may involve shared controls accross studies, ie non-zero off-diagonal elements of the matrix N00.

The output standard errors are approximate (based on inverting p-values) and are used for constructing confidence intervals in `h.summary` and `h.forestPlot`.

Currently ASSET calculates p-values by a stochastic approximation to the DLM formula as described in Bhattacharjee et al. (In Preparation). The method works by simulating truncated multivariate normal variates by importance sampling to estimate the probability term appearing in the DLM formula. Since version 2.0.0, the previous `meth.pval="DLM"` option to calculate upper bound p-values (as in Bhattacharjee et al. 2012) has been dropped as the current stochastic approximation is expected to be more accurate in all cases although slightly slower. The new p-value method also enables pre-screening of traits by the `p.bound` argument.

Specifying a p-value upper bound through `p.bound`, helps in speeding up the code when the number of traits or subtypes is relatively large. For example if `p.bound=0.25` is chosen, on an average (under the null) only a quarter of the traits will be included in subset search, allowing more traits to be analyzed in a computationally feasible manner. Note that the studies being maximized over will vary from SNP to SNP, and appropriate multiple-testing adjustment is done internally to account for this pre-selection.

The arguments `NSAMP` and `NSAMP0` give the number of importance sampling replicates to be generated. Either of these can be increased to achieve more accuracy at the cost of computational speed or vice versa.

Value

A list containing 3 main component lists named:

(1) "Meta" (Results from standard fixed effect meta-analysis of all studies/traits). This list is non-null when `meta` is TRUE and contains 3 vectors named (pval, beta, sd) of length same as snp.vars.

(2) "Subset.1sided" (one-sided subset search): This list is non-null when `search` is NULL or 1 and contains, 4 vectors named (pval, beta, sd, sd.meta) of length same as snp.vars and a logical matrix named "pheno" with one row for each snp and one column for each phenotype. For a particular SNP and phenotype, the entry has "TRUE" if this phenotype was in the selected subset for that SNP. In the output, the p-value is automatically adjusted for multiple testing due to subset search. The beta and sd correspond to the standard fixed-effect meta-analysis estimate and corresponding standard error estimate for the regression coefficient of a SNP based only on those studies/traits that are included in the identified subset. The vector sd.meta gives the meta-analysis standard errors for estimates of beta based on studies in the identified subset ignoring the randomness of the subset.

(3) Subset.2sided (two-sided subset search) This list is non-null when `search` is NULL or 2 and contains 9 vectors named (pval, pval.1, pval.2, beta.1, sd.1, beta.2, sd.2, sd.1.meta, sd.2.meta) of length same as snp.vars and two matrices named "pheno.1" and "pheno.2" giving logical indicators of a phenotype being among the positively or negatively associated subsets (respectively) as identified by 2-sided subset search. In the output, while pval provides the significance of the overall test-statistics that combined association signals from two directions, pval.1 and and pval.2 return the corresponding level of significance for each of the component one-sided test-statistics in the positive and negative directions. The values (beta.1, sd.1, beta.2, sd.2) denote the corresponding meta-analysis estimate of regression coefficients and standard errors for the identifed subsets of traits/studies that show association in positive and negative directions, respectively. The vector

sd.1.meta and sd.2.meta give the meta-analysis standard errors for estimates of beta based on studies in the identified subset ignoring the randomness of the subset in positive and negative directions, respectively.

The other objects in the list are the input arguments passed into `h.traits`.

References

Samsiddhi Bhattacharjee, Preetha Rajaraman, Kevin B. Jacobs, William A. Wheeler, Beatrice S. Melin, Patricia Hartge, GliomaScan Consortium, Meredith Yeager, Charles C. Chung, Stephen J. Chanock, Nilanjan Chatterjee. A subset-based approach improves power and interpretation for combined-analysis of genetic association studies of heterogeneous traits. Am J Hum Genet, 2012, 90(5):821-35

Bhattacharjee et al. Pre-screening and Meta-analysis based on Subset: A Fast and Powerful Approach to Pleitropic Analysis Across a Large Number of Traits (In Preparation)

See Also

`h.summary`, `h.forestPlot`

Examples

```
# Use the example data
data(ex_trait, package="ASSET")

# Display the data, and case/control overlap matrices
data
N00
N11
N10

# Define the input arguments to h.traits
snps       <- as.vector(data[, "SNP"])
traits.lab <- paste("Trait_", 1:6, sep="")
beta.hat   <- as.matrix(data[, paste(traits.lab, ".Beta", sep="")])
sigma.hat  <- as.matrix(data[, paste(traits.lab, ".SE", sep="")])
cor        <- list(N11=N11, N00=N00, N10=N10)
ncase      <- diag(N11)
ncntl      <- diag(N00)

# Now let us call h.traits on these summary data.
res <- h.traits(snps, traits.lab, beta.hat, sigma.hat, ncase=ncase,
                ncntl=ncntl, cor=cor, cor.numr=FALSE, search=NULL,
                side=2, meta=TRUE, zmax.args=NULL)

h.summary(res)
```

`h.types` *Heterogeneous Subtype analysis*

Description

Subset-based analysis of case-control studies with heterogeneous disease subtypes.

Usage

```
h.types(dat, response.var, snp.vars, adj.vars, types.lab, cntl.lab,
        subset=NULL, method=NULL, side=2, logit=FALSE, test.type="Score",
        zmax.args=NULL, pval.args=NULL, p.bound = 1,
        NSAMP=5000, NSAMP0=50000)
```

Arguments

`dat` A data frame containing individual level data for phenotype (disease status/subtype information), covariate data and SNPs. No default.

`response.var` Variable name or position of the response variable column in the data frame. This variable needs to contain disease status/subtype information in the data frame. No default.

`snp.vars` A character or numeric vector giving the variable names or positions of the SNP variables. Missing values for SNP genotypes are indicated by NA. No default.

`adj.vars` A character or numeric vector containing the variable names or positions of the columns in the data frame that would be used as adjusting covariates in the analysis. Use NULL if no covariates are used for adjustment.

`types.lab` NULL or a character vector giving the names/identifiers of the disease subtypes in `response.var` to be included in the analysis. If NULL, then all subtypes will be included. No default.

`cntl.lab` A single character string giving the name/identifier of controls (disease-free subjects) in `response.var`. No default.

`subset` A logical vector with length=nrow(`dat`) indicating the subset of rows of the data frame to be included in the analysis. Default is NULL, all rows are used.

`method` A single character string indicating the choice of method as "case-control" or "case-complement". The Default option is NULL which will carry out both types of analysis. For the case-complement analysis of disease subtype `i`, the set of control subjects is formed by taking the complement of disease subtype `i`, ie the original controls and the cases not defined by disease subtype `i`.

`side` A numeric value of either 1 or 2 indicating whether one or two-tailed p-values should be computed, respectively. The default is 2.

`logit` If TRUE, results are returned from an overall case-control analysis using standard logistic regression. Default is FALSE.

`test.type` A character string indicating the type of tests to be performed. The current options are "Score" and "Wald". The default is "Score."

`zmax.args` Optional arguments to be passed to `z.max` as a named list. This option can be useful if the user wants to restrict subset searches in some structured way, for example, incorporating ordering constraints.

`pval.args` Optional arguments to be passed to `p.dlm` as a named list. This option can be useful if the user wants to restrict subset searches in some structured way, for example, incorporating ordering constraints.

`p.bound` P-value threshold for screening studies based on marginal association before performing subset search. Default is 1, that is all studies are included in the subset search. The p-value for the overall procedure accounts for this pre-screening step. See details.

`NSAMP` Number of samples from a truncated multivariate normal distribution used to compute the DLM p-value. The default is 5000.

`NSAMP0` Number of samples from truncated multivariate normal distribution used to calculate the probability of the truncation region in DLM p-value calculation. For 1-sided subset search this is ignored unless `p.bound` < 1. The default is 50000. See details.

Details

The output standard errors are approximate (based on inverting DLM pvalues) and are used for constructing confidence intervals in `h.summary` and `h.forestPlot`. For a particular SNP, if any of the genotypes are missing, then those subjects will be removed from the analysis for that SNP.

Currently ASSET calculates p-values by a stochastic approximation to the DLM formula as described in Bhattacharjee et al. (In Preparation). The method works by simulating truncated multivariate normal variates by importance sampling to estimate the probability term appearing in the DLM formula. Since version 2.0.0, the previous `meth.pval="DLM"` option to calculate upper bound p-values (as in Bhattacharjee et al. 2012) has been dropped as the current stochastic approximation is expected to be more accurate in all cases although slightly slower. The new p-value method also enables pre-screening of traits by the `p.bound` argument.

Specifying a p-value upper bound through `p.bound`, helps in speeding up the code when the number of traits or subtypes is relatively large. For example if `p.bound=0.25` is chosen, on an average (under the null) only a quarter of the traits will be used for subset search, allowing more traits to be analyzed in a computationally feasible manner.

The arguments `NSAMP` and `NSAMP0` give the number of importance sampling replicates to be generated. Either of these can be increased to achieve more accuracy at the cost of computational speed or vice versa.

Value

A list containing 3 component lists named:

(1) "Overall.Logistic" (output for overall case-control analysis using standard logistic regression): This list is non-null when `logit` is TRUE and contains 3 vectors named (pval, beta, sd) of length same as snp.vars.

(2) "Subset.Case.Control" (output for subset-based case-control analysis): This list is non-null when `method` is NULL or "case-control". The output contains, 3 vectors named (pval, beta, sd) of length same as snp.vars and a logical matrix named "pheno" with one row for each snp and one column for each disease subtype. For a particular SNP and disease-subtype, the corresponding entry is "TRUE" if that disease subtype is included the best subset of disease subtypes that is identified to be associated with the SNP in the subset-based case-control analysis. In the output, the p-value is automatically adjusted for multiple testing due to subset search. The beta and sd corresponds to estimate of log-odds-ratio and standard error for a SNP from a logistic regression analysis involving the cases of the identified disease subtypes and the controls.

(3) "Subset.Case.Complement" (output for subset-based case-complement analysis): This list is non-null when `method` is NULL or "case-complement". The output contains, 3 vectors named (pval, beta, sd) of length same as snp.vars and a logical matrix named "pheno" with one row for each snp and one column for each disease subtype. For a particular SNP and disease-subtype, the corresponding entry is "TRUE" if that disease subtype is included the best subset of disease subtypes that is identified to be associated with the SNP in the subset-based case-complement analysis. In the output, the p-value is automatically adjusted for multiple testing due to subset search. The beta and sd corresponds to estimate of log-odds-ratio and standard error for the SNP from a logistic regression analysis involving the cases of the selected disease subtypes and the whole complement set of subjects that includes original controls and the cases of unselected disease subtypes.

References

Samsiddhi Bhattacharjee, Preetha Rajaraman, Kevin B. Jacobs, William A. Wheeler, Beatrice S. Melin, Patricia Hartge, GliomaScan Consortium, Meredith Yeager, Charles C. Chung, Stephen J. Chanock, Nilanjan Chatterjee. A subset-based approach improves power and interpretation for combined-analysis of genetic association studies of heterogeneous traits. Am J Hum Genet, 2012, 90(5):821-35

See Also

h.summary, h.forestPlot

Examples

```
# Use the example data
data(ex_types, package="ASSET")

# Display the first 10 rows of the data and a table of the subtypes
data[1:10, ]
table(data[, "TYPE"])

# Define the input arguments to h.types.
snps     <- paste("SNP_", 1:3, sep="")
adj.vars <- c("CENTER_1", "CENTER_2", "CENTER_3")
types <- paste("SUBTYPE_", 1:5, sep="")

# SUBTYPE_0 will denote the controls
res <- h.types(data, "TYPE", snps, adj.vars, types, "SUBTYPE_0", subset=NULL,
       method="case-control", side=2, logit=FALSE, test.type="Score",
       zmax.args=NULL, pval.args=NULL)


h.summary(res)
```

p.dlm *Discrete Local Maxima approximate p-value.*

Description

Function to obtain Discrete Local Maxima based estimates of p-values for z-scores maximized over subsets (of traits or subtypes), with possible restrictions and weights. Should not be called directly. See details.

Usage

```
p.dlm(t.vec, z.sub, search, side, cor.def=NULL, cor.args=NULL, sizes=NULL, p.bound=1,
      sub.def=NULL, sub.args=NULL, NSAMP=5000, NSAMP0=5e4)
```

Arguments

`t.vec` Numeric vector of (positive) points for which to calculate p-values, i.e. general observed Z-max values. No default.

`z.sub` Integer vector of the studies (traits) or subtypes being analyzed. No default.

`search` 0, 1 or 2. Search option, with 0 indicating subtype analysis, 1 and 2 denote one-sided and two-sided subset-search. No default.

`side` Either 1 or 2. For two-tailed tests (where absolute values of Z-scores are maximized), `side` should be 2. For one-tailed tests, `side` should be 1 (positive tail assumed). No default. Ignored when search is 2.

`cor.def` A function with at least 3 arguments which calculates correlation between its first argument (a subset) and its second argument (subsets such as its neighbors). The third argument is the number of traits/subtypes and the function should return a vector of correlations with the neighbors. If NULL or a non-function value is specified, internal default functions for the corresponding search option are used.

`cor.args` Other arguments to be passed to `cor.def`. These can include sample sizes and overlaps of different studies or subtypes and analysis option such as case-control or case-complement that affect the correlation structure. If `cor.def` is NULL, then `ncase` and `ncntl` must be specified in this list.

`sizes` Sizes of equivalence classes of traits. By default, no two traits or studies are equivalent. This argument is for internal use.

`p.bound` Maximum p-value above which studies are not considered in the maximization. Default is 1. See details.

`sub.def` A function to restrict subsets, e.g., order restrictions in subtype analysis. Should accept a subset (a logical vector of size k) as its first argument and should return TRUE if the subset satisfies restrictions and FALSE otherwise. Default is NULL implying all (2^k - 1) subsets are considered in the maximum.

`sub.args` Other arguments to be passed to `sub.def` as list. Default is NULL (i.e. none).

`NSAMP` Number of samples from a truncated multivariate normal distribution used to compute the DLM p-value. The default is 5000.

`NSAMP0` Number of samples from truncated multivariate normal distribution used to calculate the probability of the truncation region in DLM p-value calculation. For 1-sided subset search this is ignored unless `p.bound` < 1. The default is 50000.

Details

The function is vectorized to handle blocks of SNPs at a time. This is a helper function that is called internally by `h.traits` and `h.types` and should not be called directly. The arguments of this function that have defaults, e.g. `sub.def` can be customized using the argument `pval.args` in `h.traits` and `h.types`. Specifying a p-value upper bound through `p.bound`, helps in speeding up the code when the number of traits or subtypes is relatively large. For example if `p.bound=0.25` is chosen, on an average (under the null) only a quarter of the traits will be maximized, allowing more traits to be analyzed in a computationally feasible manner.

Note that currently the DLM p-values are stochastic in nature (based on importance sampling). To get replicable results `set.seed` can be used.

Value

A numeric vector of estimated p-values.

Examples

```
# A function to define the correlations between a subset and its neighbors
# Returned values should not exceed the value of 1
cor.def <- function(subset, neighbors, k, ncase, ncntl) {
  n <- ncol(neighbors)
  mat <- matrix(subset, nrow=k, ncol=n, byrow=FALSE)
  cor <- (mat + neighbors)*(1:k)/(k^2)
  cor <- colSums(cor)
  cor <- cor/max(cor)
  dim(cor) <- c(n, 1)

  cor
}

# Subset definition
sub.def <- function(logicalVec, args) {
  # Only allow the cummulative subsets:
  # TRUE FALSE FALSE FALSE ...
  # TRUE TRUE FALSE FALSE ...
  # TRUE TRUE TRUE FALSE ...
  # etc
  sum <- sum(logicalVec)
  ret <- all(logicalVec[1:sum])

  ret
}

k     <- 5
t.vec <- 1:k
z.sub <- rep(1, k)

p.dlm(t.vec, z.sub, 1, 2, cor.def=cor.def, sub.def=sub.def,
       cor.args=list(ncase=rep(1000, k), ncntl=rep(1000,k)))
```

z.max *Z-score Maximization*

Description

Function to maximize z-scores over subsets of traits or subtypes, with possible restrictions and weights. Should not be called directly. See details.

Usage

```
z.max(k, snp.vars, side, meta.def, meta.args, th=NULL, z.sub=rep(1, k),
      sub.def=NULL, sub.args=NULL)
```

Arguments

k Single integer. The total number of traits or studies or subtypes being analyzed. No default snp.vars Vector of integers or string labels for the SNPs being analyzed. No default.

side Either 1 or 2. For two-tailed tests (where absolute values of Z-scores are maximzed), side should be 2. For one-tailed tests, side should be 1 (positive tail is assumed). Default is 2, ignored when search is 2.

meta.def Function that calculates z-scores for a given subset, for all the SNPs. Should accept a subset (logical vector of length k) as its first argument, followed by a list of SNPs (subset of snp.vars) as its second argument. Should return a named list with at least the name "z", which is the vector of z-scores. The length of the vector should be the same length as snp.vars. Missing z-scores if any are treated as zero, in the maximization. No default.

meta.args Other arguments to be passed to meta.def as a named list. For example, this could include an entire data frame containing individual level data as in case of subtype analysis, or sample sizes and correlation matrix in case of meta-analysis of heterogeneous traits. No default.

th A vector of thresholds for each SNP, beyond which to stop maximization for that SNP. Default is a threshold of -1 for each SNP , implying no threshold. This argument is for internal use.

sub.def A function to restrict subsets, e.g., order restrictions in subtype analysis. Should accept a subset (a logical vector of size k) as its first argument and should return TRUE if the subset satisfies restrictions and FALSE otherwise. Default is NULL implying all (2^k - 1) subsets are considered in the maximum.

sub.args Other arguments to be passed to sub.def as list. Default is NULL (i.e. none).

z.sub Subset of traits/subtypes over whose subsets the maximization should be restricted. Default is all traits/subtypes. (i.e. none).

Details

This function loops through all possible (2^k - 1) subsets of (k) studies (or traits or subtypes), skips subsets that are not valid (e.g. that do not satisfy order restrictions), and maximizes the z-scores or re-weighted z-scores if weights are specified. The function is vectorized to handle blocks of SNPs at a time. This is a helper function that is called internally by h.traits and h.types and should not be called directly. The arguments of this function that have defaults, can be customized using the argument zmax.args in h.traits and h.types. Specifying a a subset of traits/subtypes z.sub to be considered for maximization, helps in speeding up the code when the number of traits or subtypes is relatively large. For example if p.bound=0.25 is chosen in h.traits, on an average (under the null) only a quarter of the traits will be maximized, allowing more traits to be analyzed in a computationally feasible manner. Note that the studies being maximized over will vary from SNP to SNP, and appropriate multiple-testing adjustment is done internally to account for this preselection.

Value

A list with two components. A vector of optimized z-scores (opt.z) and a logical matrix (opt.s) of dimension length(snp.vars) by k. Each row of (opt.s) has indicators of each trait/subtype being included in the best (optimal) subset.

Examples

```
set.seed(123)

# Define the function to calculate the z-scores
```

```
meta.def <- function(logicalVec, SNP.list, arg.beta, arg.sigma) {

  # Get the snps and subset to use
  beta <- as.matrix(arg.beta[SNP.list, logicalVec])
  se   <- as.matrix(arg.sigma[SNP.list, logicalVec])
  test <- (beta/se)^2
  ret  <- apply(test, 1, max)
  list(z=ret)
}

# Define the function to determine which subsets to consider
sub.def <- function(logicalVec, args) {
  # Only allow the cummulative subsets:
  # TRUE FALSE FALSE FALSE ...
  # TRUE TRUE FALSE FALSE ...
  # TRUE TRUE TRUE FALSE ...
  # etc
  sum <- sum(logicalVec)
  ret <- all(logicalVec[1:sum])
  ret
}

# Assume there are 10 subtypes and 3 SNPs
k        <- 10
snp.vars <- 1:3

# Generate some data
nsnp     <- length(snp.vars)
beta     <- matrix(-0.5 + runif(k*nsnp), nrow=nsnp)
sigma    <- matrix(runif(k*nsnp)^2, nrow=nsnp)

meta.args <- list(arg.beta=beta, arg.sigma=sigma)

z.max(k, snp.vars, 2, meta.def, meta.args, sub.def=sub.def)
```

Index