Package ‘caseControlGE’

May 29, 2018

Type Package

Title Semiparametric Gene-Environment Interactions in Case-Control Studies

Version 0.1.16

Author Alex Asher

Maintainer Alex Asher <alexasher@stat.tamu.edu>

Description Implements the methods of Stalder et. al. (https://doi.org/10.1093/biomet/asx045) and Wang et. al. (forthcoming). These are retrospective estimators that assume Gene-Environment independance in the source population, but place no assumptions on the marginal distributions of genetic or environmental variables. G and E can be multivariate, and both continuous and discrete variables may be used.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports Rcpp (>= 0.12.16), ucminf (>= 1.1-3)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

NeedsCompilation yes

R topics documented:

caseControlGE-package *Semiparametric Gene-Environment Interactions in Case-Control Studies*

Description

An R package for analysis of gene-environment interactions in case-control studies, using distribution-free retrospective methodology. The method of Stalder et. al. (2017) and the improvement suggested by Wang et. al (2018) use a retrospective likelihood framework under the assumption of gene-environment independence (in the population) to gain efficiency when estimating the interaction effects of genetic and environmental variables.

Details

Both methods treat the genetic and environmental variables nonparametrically, facilitating the analysis of polygenic risk factors for which distributional assumptions are difficult to justify.

Keywords

case-control study; gene-environment interaction; genetic epidemiology; retrospective method; semiparametric analysis; pseudolikelihood; polygenic analysis

Contents

- `spmle` implements method of Stalder et. al. (2017). Given binary response `D` (disease status), a vector or matrix of genetic risk factors `G`, a vector or matrix of environmental risks `E`, and the population disease rate `pi1`, `spmle` fits a model of the form `D ~ G * E` by maximizing the retrospective pseudolikelihood.
- `spmleCombo` implements the method of of Wang et. al (2018) under the same set of assumptions as `spmle`. This function takes the same imput as `spmle` with the addition of `nboot` (number of bootstrap samples) and `ncores` (number of CPU cores to use simultaneously). `spmleCombo` produces estimates that, on average, have significantly smaller mean squared error than `spmle`, at the cost of increased computation to calculate the bootstrap standard error.
- `simulateCC` simulates case-control data with a wide range of possible genetic and environmental variables.
- **methods for class** `"spmle"` both `spmleCombo` and `spmle` return objects of class `"spmle"`. A range of S3 methods are provided: `anova.spmle`, `confint.spmle`, `logLik.spmle`, `model.matrix.spmle`, `plot.spmle`, `predict.spmle`, `print.spmle`, `print.summary.spmle`, `summary.spmle`, `vcov.spmle`.

References

Stalder, O., Asher, A., Liang, L., Carroll, R. J., Ma, Y., and Chatterjee, N. (2017). *Semi-parametric analysis of complex polygenic gene-environment interactions in case-control studies.* Biometrika, 104, 801–812.

Wang, T., Asher, A., Carroll, R. J. (2018). *Improved Semiparametric Analysis of Polygenic Gene-Environment Interactions in Case-Control Studies* Unpublished.

`predict.spmle` *Predict method for spmle objects*

Description

Obtains predictions from a fitted spmle object.

Usage

```
## S3 method for class 'spmle'
predict(object, newdata, se.fit = FALSE,
  interval = c("none", "confidence"), level = 0.95, type = c("link",
  "response"), na.action = na.pass, ...)
```

Arguments

`object`	of class inheriting from "spmle"
`newdata`	an optional list or data frame in which to look for variables to use when making predictions. If omitted, the fitted values are used.
`se.fit`	a switch indicating if standard errors are required.
`interval`	Type of interval calculation. Can be abbreviated. Prediction intervals are not meaningful for binary responses and are not allowed.
`level`	confidence level.
`type`	the type of prediction required. The default is `"link"` which uses the logit scale of the linear predictors, giving the log odds. The alternative `"response"` uses the probability scale, giving `Pr(D=1|G,E)`.
`na.action`	function determining what should be done with missing values in newdata. The default is to predict NA.
`...`	further arguments passed to or from other methods.

Details

`predict.spmle` produces predicted values, obtained by evaluating the `spmle` function in the frame newdata (which defaults to `model.frame(object)`). If the logical `se.fit` is `TRUE`, standard errors of the predictions are calculated (only in the link scale). Setting `interval="confidence"` specifies computation of confidence intervals at the specified level.

Value

`predict.spmle` produces a vector of predictions or a matrix of predictions and bounds with column names fit, lwr, and upr if interval is set.

If `se.fit` is `TRUE`, a list with the following components is returned:

`fit` vector or matrix as above

`se.fit` vector with standard error of predicted means, in the link scale `residual.scale` residual standard deviation

`df` degrees of freedom for residual

`simulateCC` *Simulate case-control data with multivariate, possibly dependent genetic and environmental components.*

Description

`simulateCC` simulates case-control data to be analyzed by `spmle`, `spmleCombo`, logistic regression, or other methods.

Usage

```
simulateCC(ncase, ncontrol, beta0, betaG_SNP, betaG_normPRS, betaG_gammaPRS,
  betaG_bimodalPRS, betaE_bin, betaE_norm, betaGE_SNP_bin, betaGE_normPRS_bin,
  betaGE_gammaPRS_bin, betaGE_bimodalPRS_bin, betaGE_SNP_norm,
  betaGE_normPRS_norm, betaGE_gammaPRS_norm, betaGE_bimodalPRS_norm, MAF,
  SNP_cor = 0, G_normPRS_cor = 0, E_norm_cor = 0, E_bin_freq,
  regress_E_bin_on_G_SNP, regress_E_bin_on_G_normPRS,
  regress_E_bin_on_G_gammaPRS, regress_E_bin_on_G_bimodalPRS,
  regress_E_norm_on_G_SNP, regress_E_norm_on_G_normPRS,
  regress_E_norm_on_G_gammaPRS, regress_E_norm_on_G_bimodalPRS,
  control = list())
```

Arguments

`ncase, ncontrol`
: number of cases and controls, both must be positive integers.

`beta0`
: logistic intercept, required. Can be manipulated to change the population disease rate.

`betaG_SNP, betaG_normPRS, betaG_gammaPRS, betaG_bimodalPRS`
: optional coefficients for genetic main effects (at least one must be specified). Genetic variables can include SNPs and polygenic risk scores with normal, gamma, and bimodal distributions. Vector valued coefficients produce multivariate genetic data. When simulating SNPs, you must provide `MAF` with the same length as `betaG_SNP`.

`betaE_bin, betaE_norm`
: optional coefficients for environmental variable main effects (at least one must be specified). Environmental variables can include binary and normally distributed random variables. Vector valued coefficients produce multivariate environmental data.

`betaGE_SNP_bin, betaGE_normPRS_bin, betaGE_gammaPRS_bin, betaGE_bimodalPRS_bin`
: ,

`betaGE_SNP_norm, betaGE_normPRS_norm, betaGE_gammaPRS_norm, betaGE_bimodalPRS_norm`
: coefficients for multiplicative G*E interaction effects. The length of the coefficient of any given G*E interaction must equal the product of the lengths of the coefficients of the corresponding G and E main effects.

`MAF`
: Minor Allele Frequency of SNPs. This vector is the same length as `beta_G_SNP` and has values between 0 and 1. The MAF is used to generate SNP data that is in Hardy-Weinberg Equilibrium. This specifies `Pr[G=(0, 1, 2)] = [(1-MAF)^2, 2*MAF(1-MAF), MAF^2]`.

`SNP_cor`
: scalar specifying the correlation between adjacent SNPs. SNPs are simulated by generating multivariate normal random draws with an AR1(`SNP_cor`) covariance matrix. These normal draws are then trichotomized according to HWE to simulate SNPs. Default `SNP_cor = 0`.

`G_normPRS_cor`
: correlation matrix for multivariate normal polygenic risk scores. In the bivariate case, a 2x2 matrix or a scalar (for correlation) are accepted. Default `G_normPRS_cor = 0` generates independent normal polygenic risk scores.

`E_norm_cor`
: correlation matrix for multivariate normal environmental variable. In the bivariate case, a 2x2 matrix or a scalar (for correlation) are accepted. Default `E_norm_cor = 0` generates independent normal environmental variables.

`E_bin_freq`
: marginal probability that E_bin = 1. Must have length equal to `length(betaE_bin)` and values between 0 and 1.

`regress_E_bin_on_G_SNP, regress_E_bin_on_G_normPRS, regress_E_bin_on_G_gammaPRS , regress_E_bin_on_G_bimodalPRS`
: allow the simulation of case-control data that violates the G-E independence assumption. If specified, binary environmental variables will be generated with `Pr(E=1|G) = plogis[regress_E_bin_on_G * G + qlogis(E_bin_freq)]`. If these arguments are missing, `NULL`, or all `0`s, the binary environmental variables will be independent of the genetic variables. If specified, the length of the regression argument must equal the product of the lengths of the coefficients of the corresponding G and E main effects.

`regress_E_norm_on_G_SNP, regress_E_norm_on_G_normPRS, regress_E_norm_on_G_gammaPRS , regress_E_norm_on_G_bimodalPRS`
: allow the simulation of case-control data that violates the G-E independence assumption. If specified, normal environmental variables will be generated from a Normal(`regress_E_norm_on_G * G`, 1) distribution. If these arguments are missing, `NULL`, or all `0`s, the normal environmental variables will be independent of the genetic variables. If specified, the length of the regression argument must equal the product of the lengths of the coefficients of the corresponding G and E main effects.

`control`
: a list of control parameters, all of which are ignored except `trace`, a scalar. If `trace > -1`, information about the simulation (e.g. population disease rate, correlations between SNPs, etc.) is produced. Default `trace=0`.

Details

The user can specify up to four types of genetic variables, each of which can be multivariate: SNPs with additive effects under Hardy-Weinberg Equilibrium and polygenic risk scores with normal, gamma, and bimodal distributions. Two types of environmental variables (binary and normal) can also be potentially multivariate.

SNPs may be generated in linkage disequilibrium, yielding correlated SNPs. Multivariate normal polygenic risk scores may have a user-specified correlation matrix, as may multivariate normal environmental variables. Correlation may also be specified between genetic and environmental variables to simulate data in violation of the gene-environment independence assumption.

The number of variables generated is determined by the length of the betas given. If you specify `betaG_normPRS = c(log(1.1), log(1.2))` and `betaE_bin = c(log(1.2), log(1.2), log(1.2))`, you will get two `G` variables (normally distributed polygenic risk scores), and three `E` variables (with binary distributions). In this example, you would supply a vector of length 6 for `betaGE_normPRS_bin`.

If both G and E are multivariate, `beta_GE_` and `regress_E_` arguments iterate G quickly and E slowly. In the example above, `betaGE_normPRS_bin` is ordered (G1*E1, G2*E1, G1*E2, G2*E2, G1*E3, G2*E3).

Value

`simulateCC` produces a list with three elements:

D	a binary vector with `ncontrol` zeros and `ncase` ones.
G	a matrix with `ncontrol + ncase` rows and a column for each genetic variable. Genetic variables are ordered: SNPs, normal PRSs, gamma PRSs, bimodal PRSs.
E	a matrix with `ncontrol + ncase` rows and a column for each environmental variable. Environmental variables are ordered: binary, normal.

See Also

`spmleCombo`, `spmle`

Examples

```
set.seed(2018)
# Simulation from Table 1 in Stalder et. al. (2017)
dat = simulateCC(ncase=1000,
                 ncontrol=1000,
                 beta0=-4.165,
                 betaG_SNP=c(log(1.2), log(1.2), 0, log(1.2), 0),
                 betaE_bin=log(1.5),
                 betaGE_SNP_bin=c(log(1.3), 0, 0, log(1.3), 0),
                 MAF=c(0.1, 0.3, 0.3, 0.3, 0.1),
                 SNP_cor=0.7,
                 E_bin_freq=0.5)

# Simulation with 5 SNPs and a single normal environmental variable
# that is dependent on G1 with an R^2 of 0.001.
# True population disease rate in this simulation is 0.03.
# This simulation scenario was used in the Supplementary Material of Stalder et. al. (2017)
dat2 = simulateCC(ncase=1000,
                  ncontrol=1000,
                  beta0=-3.89,
                  betaG_SNP=c(log(1.2), log(1.2), 0, log(1.2), 0),
                  betaE_norm=log(1.5)/(qnorm(0.75)-qnorm(0.25)),
               betaGE_SNP_norm=c(log(1.3), 0, 0, log(1.3), 0) / (qnorm(0.75)-qnorm(0.25)),
                  MAF=c(0.1, 0.3, 0.3, 0.3, 0.1),
                  SNP_cor=0.7,
                  regress_E_norm_on_G_SNP=c(sqrt(0.001),rep(0,4)),
                  control=list(trace=1))
```

spmle — *Semiparametric Maximum Pseudolieklihood Estimator for Case-Control Studies Under G-E Independence.*

Description

`spmle` maximizes the retrospective pseudolikelihood of case-control data under the assumption of G-E independence in the underlying population. The marginal distributions of G and E are treated nonparametrically.

Usage

```
spmle(D, G, E, pi1, data, control = list(), swap = FALSE, startvals)
```

Arguments

`D`	a binary vector of disease status (1=case, 0=control).
`G`	a vector or matrix (if multivariate) containing genetic data. Can be continuous, discrete, or a combination.
`E`	a vector or matrix (if multivariate) containing environmental data. Can be continuous, discrete, or a combination.
`pi1`	the population disease rate, a scalar in [0, 1) or the string "rare". Using `pi1=0` is the rare disease approximation.
`data`	an optional list or environment containing the variables in the model. If not found in `data`, the variables are taken from the environment from which `spmle` is called.
`control`	a list of control parameters that allow the user to control the optimization algorithm. See 'Details'.
`swap`	a logical scalar rarely of interest to the end user. Dependence on the distributions of G and E are removed using different methods; this switch swaps them to produce a symmetric estimator with identical properties to the SPMLE. Default `FALSE`.
`startvals`	an optional numeric vector of coefficient starting values for optimization. Usually left blank, in which case logistic regression estimates are used as starting values.

Details

This function applies the method of Stalder et. al. (2017) to maximize the retrospective pseudolikelihood of case-control data under the assumption of G-E independence. It currently supports the model with G and E main effects and a multiplicative G*E interaction.

The `control` argument is a list that controls the behavior of the optimization algorithm `ucminf` from the **ucminf** package. When `ucminf` works, it works brilliantly (typically more than twice as fast as the next-fastest algorithm). But it has a nasty habit of declaring convergence before actually converging. To address this, `spmle` checks the maximum gradient at "convergence", and can rerun the optimization using different starting values. The `control` argument can supply any of the following components:

`max_grad_tol` maximum allowable gradient at convergence. `spmle` does not consider the optimization to have converged if the maximum gradient `>` `max_grad_tol` when `ucminf` stops. Default `max_grad_tol = 0.001`.

`num_retries` number of times to retry optimization. An error is produced if the optimization has not converged after `num_retries`. Different starting values are used for each retry. Default `num_retries = 2`.

`use_hess` a logical value instructing `spmle` to use the analytic hessian to precondition the optimization. This brings significant speed benefits, and is one reason `ucminf` is so fast. For unknown reasons, preconditioning causes computers with certain Intel CPUs to prematurely terminate iterating. By default, `use_hess = TRUE`, but if you notice that `ucminf` never converges during the first attempt, try setting `use_hess = FALSE`.

`trace` a scalar or logical value that is used by both `spmle` and `ucminf` to control the printing of detailed tracing information. If TRUE or > 0, details of each `ucminf` iteration are printed. If FALSE or 0, `ucminf` iteration details are suppressed but `spmle` still prints optimization retries. If `trace < 0` nothing is printed. Default `trace = 0`.

additional control parameters not used by `spmle`, but are passed to `ucminf`. Note that the `ucminf` algorithm has four stopping criteria, and `ucminf` will declare convergence if any one of them has been met. The `ucminf` control parameter `"grtol"` controls `ucminf`'s gradient stopping criterion, which defaults to `1e-6`. `grtol` should not be set larger than the `spmle` control parameter `max_grad_tol`.

Value

an object of class `"spmle"`. The function `summary` (i.e., `summary.spmle`) can be used to obtain or print a summary of the results.

The function `anova` (i.e., `anova.spmle`) will conduct likelihood-ratio tests comparing one `spmle` object to another. These are valid tests because the loglikelihood reported by `logLik.spmle` is accurate up to an additive constant. However `anova` should not be used to compare an `spmle` object to a model fit by a different method.

`predict.spmle`, the `predict` method for S3 class `"spmle"`, can predict the expected response (on logistic or probability scales), compute confidence intervals for the expected response, and provide standard errors.

The generic accessor functions `coefficients`, `fitted.values` and `residuals` can be used to extract various useful features of the value returned by `spmle`.

An object of class `"spmle"` is a list containing at least the following components:

`coefficients` a named vector of coefficients

`pi1` the value of `pi1` used during the analysis

`SE` standard error estimate of coefficients

`cov` estimated covariance matrix of coefficients

`glm_fit` a logistic regression model fit using the same model as `spmle`

`call` the matched call

`formula` the formula used

`data` the `data argument`

`model` the model frame

`terms` the `terms` object used

`linear.predictors` the linear fit on the logistic link scale

`fitted.values` the fitted values on the probability scale

`residuals` the Pearson residuals

`null.deviance` the deviance for the null model. Deviance = `-2*logLik`.

`df.residual` the residual degrees of freedom

`df.null` the residual degrees of freedom for the null model

`rank` the numeric rank of the fitted linear model (i.e. the number of parameters estimated)

`nobs` number of observations

`ncase` number of cases

`ncontrol` number of controls

`spmle` objects created by `spmle()` additionally have components `logLik` (log pseudolikelihood), `deviance` (-2 * log pseudolikelihood), `aic`, `bic`, `ucminf` (optimization output), and matrices `H_inv`, `Sigma`, `zeta0`, and `zeta1`, which are used in calculating the asymptotic estimate of standard error.

References

Stalder, O., Asher, A., Liang, L., Carroll, R. J., Ma, Y., and Chatterjee, N. (2017). *Semi-parametric analysis of complex polygenic gene-environment interactions in case-control studies.* Biometrika, 104, 801–812.

See Also

`spmleCombo` for a slower but more precise estimator, `simulateCC` to simulate data

Examples

```
# Simulation from Table 1 in Stalder et. al. (2017)
set.seed(2018)
dat = simulateCC(ncase=500, ncontrol=500, beta0=-4.165,
                 betaG_SNP=c(log(1.2), log(1.2), 0, log(1.2), 0),
                 betaE_bin=log(1.5),
                 betaGE_SNP_bin=c(log(1.3), 0, 0, log(1.3), 0),
                 MAF=c(0.1, 0.3, 0.3, 0.3, 0.1),
                 SNP_cor=0.7, E_bin_freq=0.5)

# SPMLE with known population disease rate of 0.03
spmle(D=D, G=G, E=E, pi1=0.03, data=dat)

# Simulation with a single SNP and a single binary environmental variable.
# True population disease rate in this simulation is 0.03.
# This simulation scenario was used in the Supplementary Material of Stalder et. al. (2017)
# to compare performance against the less flexible method of Chatterjee and Carroll (2005),
# which is available as the function as snp.logistic in the Bioconductor package CGEN.
dat2 = simulateCC(ncase=100, ncontrol=100, beta0=-3.77,
                  betaG_SNP=log(1.2), betaE_bin=log(1.5),
                  betaGE_SNP_bin=log(1.3), MAF=0.1,
                  E_bin_freq=0.5)

# SPMLE using the rare disease assumption, optimization tracing,
# and no hessian preconditioning.
spmle(D=D, G=G, E=E, pi1=0, data=dat2, control=list(trace=0, use_hess=FALSE))
```

---

spmleCombo — *Improved Semiparametric Estimator for Case-Control Studies Under G-E Independence.*

Description

`spmleCombo` estimates Gene-Environment interactions in case-control data under the assumption of G-E independence in the underlying population. This is an improved version of `spmle` that, on average, has significantly smaller mean squared error than `spmle`, at the cost of increased computing time.

Usage

```
spmleCombo(D, G, E, pi1, data, nboot = 50, ncores = 1, control = list(),
  startvals)
```

Arguments

`D`	a binary vector of disease status (1=case, 0=control).
`G`	a vector or matrix (if multivariate) containing genetic data. Can be continuous, discrete, or a combination.
`E`	a vector or matrix (if multivariate) containing environmental data. Can be continuous, discrete, or a combination.
`pi1`	the population disease rate, a scalar in [0, 1) or the string "rare". Using `pi1=0` is the rare disease approximation.
`data`	an optional list or environment containing the variables in the model. If not found in `data`, the variables are taken from the environment from which `spmleCombo` is called.

`nboot`	the number of bootstraps to use when estimating the standard error, an integer. Setting `nboot=0` disables the bootstrap and uses the asymptotic standard error estimate (not recommended because the asymptotic SE often has poor coverage: setting `nboot=0` will trigger a warning). Default `nboot = 50`.
`ncores`	the number of cpu cores to use when parallelizing bootstraps, an integer. Default `ncores = 1` executes the bootstrap sequentially.
`control`	a list of control parameters that allow the user to control the optimization algorithm. See 'Details'.
`startvals`	an optional numeric vector of coefficient starting values for optimization. Usually left blank, in which case logistic regression estimates are used as starting values.

Details

This function calculates the Symmetric Combination Estimator of Wang et. al. (2018), which improves estimation efficiency in case-control studies of gene-environment interactions while treating the marginal distributions of G and E nonparametrically.

This is done by calling `spmle` twice (once with `swap=TRUE`) to generate two symmetric estimates, which are then combined using a GLS approach. It currently supports the model with G and E main effects and a multiplicative G*E interaction.

The `control` argument is a list that controls the behavior of the optimization algorithm `ucminf` from the **ucminf** package. When `ucminf` works, it works brilliantly (typically more than twice as fast as the next-fastest algorithm). But it has a nasty habit of declaring convergence before actually converging. To address this, `spmleCombo` checks the maximum gradient at "convergence", and can rerun the optimization using different starting values. The `control` argument can supply any of the following components:

`max_grad_tol` maximum allowable gradient at convergence. `spmleCombo` does not consider the optimization to have converged if the maximum gradient > `max_grad_tol` when `ucminf` stops. Default `max_grad_tol = 0.001`.

`num_retries` number of times to retry optimization. An error is produced if the optimization has not converged after `num_retries`. Different starting values are used for each retry. Default `num_retries = 2`.

`use_hess` a logical value instructing `spmleCombo` to use the analytic hessian to precondition the optimization. This brings significant speed benefits, and is one reason `ucminf` is so fast. For unknown reasons, preconditioning causes computers with certain Intel CPUs to prematurely terminate iterating. By default, `use_hess = TRUE`, but if you notice that `ucminf` never converges during the first attempt, try setting `use_hess = FALSE`.

`trace` a scalar or logical value that is used by both `spmleCombo` and `ucminf` to control the printing of detailed tracing information. If TRUE or > 0, details of each `ucminf` iteration are printed. If FALSE or 0, `ucminf` iteration details are suppressed but `spmleCombo` still prints optimization retries. If `trace < 0` nothing is printed. Default `trace = 0`.

additional control parameters not used by `spmleCombo`, but are passed to `ucminf`. Note that the `ucminf` algorithm has four stopping criteria, and `ucminf` will declare convergence if any one of them has been met. The `ucminf` control parameter "`grtol`" controls `ucminf`'s gradient stopping criterion, which defaults to `1e-6`. `grtol` should not be set larger than the `spmleCombo` control parameter `max_grad_tol`.

Value

an object of class "spmle". The function summary (i.e., summary.spmle) can be used to obtain or print a summary of the results. The Symmetric Combination Estimator is not a maximum (pseudo)likelihood estimator like spmle; it is the optimal combination of two such estimators. As such, it has no associated loglikelihood and the function anova.spmle cannot be used to compare models fit using spmleCombo.

predict.spmle, the predict method for S3 class "spmle", can predict the expected response (on logistic or probability scales), compute confidence intervals for the expected response, and provide standard errors.

The generic accessor functions coefficients, fitted.values and residuals can be used to extract various useful features of the value returned by spmleCombo.

An object of class "spmle" is a list containing at least the following components:

coefficients	a named vector of coefficients
pi1	the value of pi1 used during the analysis
SE	standard error estimate of coefficients
cov	estimated covariance matrix of coefficients
glm_fit	a logistic regression model fit using the same model as spmleCombo
call	the matched call
formula	the formula used
data	the data argument
model	the model frame
terms	the terms object used
linear.predictors	the linear fit on the logistic link scale
fitted.values	the fitted values on the probability scale
residuals	the Pearson residuals
null.deviance	the deviance for the null model
df.residual	the residual degrees of freedom
df.null	the residual degrees of freedom for the null model
rank	the numeric rank of the fitted linear model (i.e. the number of parameters estimated)
nobs	number of observations
ncase	number of cases
ncontrol	number of controls

spmle objects created by spmleCombo() additionally have components spmle_E (model from spmle that profiled out the distribution of E), spmle_G (model from spmle that profiled out the distribution of G with swap=TRUE), and bootstraps (matrix of bootstrapped parameter estimates, if nboot > 0).

References

Stalder, O., Asher, A., Liang, L., Carroll, R. J., Ma, Y., and Chatterjee, N. (2017). *Semi-parametric analysis of complex polygenic gene-environment interactions in case-control studies.* Biometrika, 104, 801–812.

Wang, T., Asher, A., Carroll, R. J. (2018). *Improved Semiparametric Analysis of Polygenic Gene-Environment Interactions in Case-Control Studies* Unpublished.

See Also

`spmle`, `simulateCC` to simulate data

Examples

```
# Simulation from Table 1 in Stalder et. al. (2017)
set.seed(2018)
dat = simulateCC(ncase=500, ncontrol=500, beta0=-4.165,
                 betaG_SNP=c(log(1.2), log(1.2), 0, log(1.2), 0),
                 betaE_bin=log(1.5),
                 betaGE_SNP_bin=c(log(1.3), 0, 0, log(1.3), 0),
                 MAF=c(0.1, 0.3, 0.3, 0.3, 0.1),
                 SNP_cor=0.7, E_bin_freq=0.5)

# SPMLE with known population disease rate of 0.03 and asymptotic SE estimates
spmleCombo(D=D, G=G, E=E, pi1=0.03, data=dat, nboot=0)
```

Index