Package: twoPhaseGAS (via r-universe)

October 31, 2026

Type Package

Title Two-Phase Genetic Association Study design and analysis with missing covariates by design

Version 1.08

Date 2021-11-01

Author Osvaldo Espin-Garcia <osvaldo.espingarcia@utoronto.ca>, Apostolos Dimitromanolakis <apostolis@live.ca>, Shelley Bull <bull@lunenfeld.ca>

Maintainer Osvaldo Espin-Garcia <osvaldo.espingarcia@utoronto.ca>

Description Provides functionality for selecting and analyzing individuals in two-phase genetic association studies. Phase 1 data usually come from GWAS results and we assume phase 2 genetic data will be part of a targeted genome sequencing / fine-mapping study. The package assists in selecting a subset of individuals that will be sequenced for phase 2. Once phase 2 data have been collected, the package implements methods to analyze phase 1 and 2 data together using semi-parametric regression models.

License GPL (>= 2)

Depends R (>= 2.15.2), stats, MASS, data.table, Matrix, dfoptim, nloptr, enrichwith

VignetteBuilder knitr

RoxygenNote 7.2.1

Suggests knitr, testthat, rmarkdown, gplots

Encoding UTF-8

Config/pak/sysreqs cmake

Repository https://egosv.r-universe.dev

Date/Publication 2022-11-06 21:52:59 UTC

RemoteUrl https://github.com/egosv/twophasegas

RemoteRef HEAD

RemoteSha ef0e7b971aa3e6d198de8592b27d77be10bc9fab

Contents

DataGeneration_TPD 2
optimTP.GA 3
optimTP.LM 4
twoPhase 6
twoPhaseDesign 7
twoPhaseHeuristic 8
twoPhaseSPML 9

Index **11**

`DataGeneration_TPD` *This code a function that generates a sample Y, Z, G given some initial parameters.*

Description

This code a function that generates a sample Y, Z, G given some initial parameters.

Usage

```
DataGeneration_TPD(
  Beta0 = 2,
  Beta1 = 0.5,
  Sigma2 = 1,
  N = 5000,
  LD.r = 0.75,
  P_g = 0.2,
  P_z = 0.3,
  tao = 2/5
)
```

Arguments

`Beta0`	intercept Default 2
`Beta1`	genetic effect Default 0.5
`Sigma2`	variance of the error term Default 1
`N`	Phase 1 sample size, i.e. GWAS data (Default: 5000)
`LD.r`	linkage disequilibrium (r) between G and Z (Default: 0.75)
`P_g`	minor allele frequency for G, the causal SNP
`P_z`	minor allele frequency for Z, the GWAS SNP
`tao`	quantile value to define the stratification for the quantitative trait (default: 2/5)

Value

A dataframe with complete data Y, G, Z, S, where G and Z come from the same haplotype determined by P_g, P_z and LD.r; Y is generated from Y = Beta0 + Beta1 x G, S is a 3 level variable determined by Y, Beta1, Sigma2 and tao. the function iterates across generated datasets until Z and Y are associated at a suggestive genome wide threshold of p<=1e-05.

Examples

```
data = DataGeneration_TPD()
```

`optimTP.GA` *function optimTP.GA*

Description

function optimTP.GA

Usage

```
optimTP.GA(
  ncores,
  formula,
  miscov,
  auxvar,
  family,
  n,
  data,
  beta,
  p_gz,
  disp = NULL,
  ga.popsize,
  ga.propelit,
  ga.proptourney,
  ga.ngen,
  ga.mutrate,
  ga.initpop = NULL,
  optimMeasure,
  K.idx = NULL,
  seed = 1,
  verbose = 0
)
```

Arguments

ncores	ncores1
formula	the formula
miscov	miscov1
auxvar	auxvar1
family	family1
n	n1
data	the data gggg
beta	beta1
p_gz	p_gz1
disp	disp1
ga.popsize	ga.popsize1
ga.propelit	ga.propelit1
ga.proptourney	ga.proptourney1
ga.ngen	ga.ngen1
ga.mutrate	ga.mutrate1
ga.initpop	ga.initpop1
optimMeasure	optimMeasure1
K.idx	K.idx1
seed	seed1
verbose	verbose1

Details

details here

Examples

```
print(1)
```

`optimTP.LM` *function optimTP.LM*

Description

function optimTP.LM

Usage

```
optimTP.LM(
  formula,
  miscov,
  auxvar,
  strata,
  family,
  n,
  data,
  beta,
  p_gz,
  disp = NULL,
  optimMeasure,
  K.idx = NULL,
  min.nk = NULL,
  logical.sub = NULL
)
```

Arguments

formula	formula1name
miscov	xxxxxxxxcode
auxvar	auxvar1
strata	strata1
family	family1
n	n1
data	data1
beta	beta1
p_gz	p_gz1
disp	disp1
optimMeasure	optimMeasure1
K.idx	K.idx1
min.nk	min.nk1
logical.sub	logical.sub1

Details

details at here

Examples

```
print(1)
```

`twoPhase` *Internal function to optimize over a range of maf, LD and betas.*

Description

Internal function to optimize over a range of maf, LD and betas.

Usage

```
twoPhase(
  beta = c(1, 1, 1),
  maf_G = 0.1,
  LD = 0.3,
  data = NA,
  n2 = NA,
  design_formula = Y ~ G + Z,
  family = gaussian(),
  useGeneticAlgorithm = FALSE
)
```

Arguments

`beta`	Vector of betas (length of 3, corresponding to intercept, effect size for G, effect size for Z).
`maf_G`	Maf for G (numeric, ranging from 0 to 1).
`LD`	Correlation r betweek G and Z (numeric, ranging from -1 to 1, r value).
`data`	Data frame with Y and Z variables.
`n2`	Phase 2 sample size
`design_formula`	Formula for the regression model, default is Y ~ G + Z. Z is the GWAS SNP, G is the sequence variant. Y is outcome.
`family`	Distrubution of the outcome (default: gaussian()). Familes available for glm can be used here. See help(stats::family) for examples.
`useGeneticAlgorithm`	If TRUE, use genetic algorithm in addition to Lagrange multiplier approach (slower). Default: FALSE

Value

sth

Examples

```
data = twoPhase()
```

`twoPhaseDesign` *Compute sample allocations for a two phase study design.*

Description

Compute sample allocations for a two phase study design.

Usage

```
twoPhaseDesign(
  beta,
  maf_G,
  LD,
  data,
  n2,
  design_formula = Y ~ G + Z,
  family = gaussian(),
  S,
  perc_Y = c(1/5, 4/5),
  p_gz,
  design = c("RDS", "LM", "GA", "PPS", "BAL", "COM", "TZL"),
  ndraws = 10,
  optimCriterion = c("Par-spec", "A-opt", "D-opt"),
  overallMethod = c("med-max", "cumm")
)
```

Arguments

`beta`	Vector of betas (length of 3, corresponding to intercept, effect size for G, effect size for Z).
`maf_G`	minor allele frequency for G (numeric, ranging from 0 to 1). Numeric or vector of possible values.
`LD`	Correlation r betweek G and Z (numeric, ranging from -1 to 1, r value). Numeric or vector of possible values.
`data`	Data frame with Y and Z variables.
`n2`	Phase 2 sample size.
`design_formula`	Formula for the regression model, default is Y ~ G + Z, where Z is the GWAS SNP, G is the sequence variant. Y is outcome. Rename the variables in your data.frame to match Y and Z, G is the seq-SNP not present in the data.frame.
`family`	Distrubution of the outcome. Default: gaussian(). Familes available for glm can be used here. See help(stats::family) for examples.
`S`	stratification of the outcome Y. Optional. Should be a numeric vector with strata categories (e.g. 1 1 2 2 3 3). If present, its length must be equal to the number of rows in data. Default: NULL. Needed when Y does not render itself into strata, e.g. Gaussian, Poisson, Gamma.

perc_Y	vector of percentiles in increasing order for which the outcome Y will be stratified. Default: c(1/5,4/5). Only used when S is NA. Note that setting up S is strongly suggested.
p_gz	data frame with the joint distribution between G and Z. See examples for the right format. Default: NULL. If present, values of maf_G and LD are disregarded in the analysis.
design	string for the design to use for phase 2 sample selection. One of residual-dependent sampling ("RDS"), optimal as defined by Tao, Zheng and Lin (2019) ("TZL"), optimal via Lagrange multipliers ("LM"), optimal via genetic algorithm ("GA"), probability proportional to size ("PPS"), balanced ("BAL") or combined ("COM") allocations. Default: "RDS". See details for a more explanations.
ndraws	integer that determines the number of draws to examine when design is one of "pps", "bal" or "comb" and the design parameter combinations is greater than 1. Default: 10
optimCriterion	string denoting the optimality criterion used during the optimization. One of "Par-spec" (default), "A-opt" or "D-opt". For parameter-specific, A-optimality or D-optimality, respectively.
overallMethod	string denoting the method to select the overall design when multiple design parameters are given. One of "med-max" (default) or "cumm" for median-maximum and cummulative frequencies, respectively. Note that in this version strata are always defined in terms of Z and S, i.e. a joint design, future implementations may relax this by allowing for only S or only Z (marginal designs, outcome- or covariate-dependent, respectively)

Value

sth

`twoPhaseHeuristic` *Select samples for phase 2 under heuristic designs.*

Description

Select samples for phase 2 under heuristic designs.

Usage

```
twoPhaseHeuristic(
  design = c("pps", "bal", "comb"),
  ndraws = 1,
  data = NA,
  n2 = NA,
  family = gaussian(),
  S = NULL,
  perc_Y = c(1/5, 4/5)
)
```

Arguments

design	Heuristic design to use for phase 2 sample selection. One of probability proportional to size ("pps"), balanced ("bal") or combined ("comb") allocations. Default: "pps".
ndraws	Number of draws of the heuristic design to generate Default: 1.
data	Data frame with Y and Z variables.
n2	Phase 2 sample size.
family	Distrubution of the outcome. Default: gaussian(). Familes available for glm can be used here. See help(stats::family) for examples.
S	stratification of the outcome Y. Optional. Should be a numeric vector with strata categories (e.g. 1 1 2 2 3 3). If present, its length must be equal to the number of rows in data. Default: NULL. Needed when Y does not render itself into strata, e.g. Gaussian, Poisson, Gamma.
perc_Y	vector of percentiles in increasing order for which the outcome Y will be stratified. Default: c(1/5,4/5). Only used when S is NA. Note that setting up S is strongly suggested. Note that in this version strata are always defined in terms of Z and S, i.e. a joint design, future implementations may relax this by allowing for only S or only Z (marginal designs, outcome- or covariate-dependent, respectively)

Value

sth

---

twoPhaseSPML *Performs inference on two-phase studies data via semiparametric maximum likelihood.*

Description

Performs inference on two-phase studies data via semiparametric maximum likelihood.

Usage

```
twoPhaseSPML(
  formula,
  miscov,
  auxvar,
  family = gaussian,
  data0,
  data1,
  start.values = NULL,
  verbose = FALSE
)
```

Arguments

formula	regression formula, note that if it does not contain the missing-by-design variable, miscov, it will return results under the null hypothesis. Hypothesis testing corresponds to the score statistic. Otherwise, estimates and hypothesis testing ocurr under the alternative hypothesis leading to Wald statistics. All the elements in formula except miscov must be present in data0 and data1.
miscov	right hand side formula with the missing-by-design covariate(s), i.e. the potential causal locus (loci). Must be present in data1 but absent in data0.
auxvar	right hand side formula with the auxiliary variable(s), i.e. the GWAS SNP from phase 1. Must be present in data0 and data1.
family	member of the exponential family (see family, 'quasi' models not available). Default gaussian().
data0	a dataframe with the complement of the phase 2 data. Must contain the unique elements in formula and auxvar but NOT miscov.
data1	a dataframe with the phase 2 data. Must contain the unique elements in formula, auxvar, and miscov.
start.values	a named list with initial values for the regression parameters and joint distribution between miscov and auxvar (only one can be specified). Defaults to NULL
verbose	verbose output? logical, defaults to FALSE.

Details

these are some additional details

Value

A list of objects

Examples

```
data = DataGeneration_TPD()
set.seed(1)
R = rep(0, nrow(data)); R[sample(nrow(data),500)] <- 1 # random phase 2 subsample of 500.
data0 = data[R==0,c('Y','Z')]
data1 = data[R==1,c('Y','Z','G1')]
res_Ho = twoPhaseSPML(formula =  Y ~ Z,
miscov = ~ G1,
auxvar = ~ Z,
data0 = data0, data1 = data1)
res_Ha = twoPhaseSPML(formula =  Y ~ Z + G1,
miscov = ~ G1,
auxvar = ~ Z,
data0 = data0, data1 = data1)
```

Index

DataGeneration_TPD, 2

family, *10*

optimTP.GA, 3
optimTP.LM, 4

twoPhase, 6
twoPhaseDesign, 7
twoPhaseHeuristic, 8
twoPhaseSPML, 9