Package ‘thgenetics’

July 22, 2025

Version 0.4-2

Date 2018-04-17

Title Genetic Rare Variants Tests

Author Thomas Hoffmann <tjhoffm@gmail.com>

Maintainer Thomas Hoffmann <tjhoffm@gmail.com>

Description A step-up test for genetic rare variants in a gene or in a pathway. The method determines an optimal grouping of rare variants analytically. The method has been described in Hoffmann TJ, Marini NJ, and Witte JS (2010) <doi:10.1371/journal.pone.0013584>.

License GPL

URL http://sites.google.com/site/thomashoffmannproject/
https://doi.org/10.1371/journal.pone.0013584

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-04-18 10:25:13 UTC

Contents

rareGeneTest *Rare Variant Tests*

Description

Runs tests for rare variants by running the "step-up" approach to choose the best set of variants (possibly signing them), and correcting by permutation. `rareGeneTest` tests a single gene, and also allows for other aggregation models described in Hoffmann et al.. `rarePathwayTest` provides an extension of this method, that does a step-up approach for inclusions of genes (for computional time). For larger genes, it may also be advantageous to run `rarePathwayTest` which will automatically chop large genes into subpieces and run step-up on those pieces.

Usage

```
rareGeneTest(genotype, phenotype,
  use_sign=TRUE, use_weight=TRUE, nperm=1000,
  binary=all(phenotype==0 | phenotype==1, na.rm=TRUE),
  strategy="step", thresh=1)
rarePathwayTest(genotype, genotype_gene, phenotype,
  use_sign=TRUE, use_weight=TRUE, nperm=1000,
  binary=all(phenotype==0 | phenotype==1, na.rm=TRUE),
  CUT=15)
```

Arguments

`genotype` Matrix of individuals along the rows and genotypes along the columns. Each entry in the matrix should be coded `NA` for missing; and `0`, `1`, or `2` for the number of alleles a person has (must be bi-allelic).

`phenotype` Vector of same length as the number of rows of the matrix. Either continuous, or dichotomous (case=`1`, control=`0`.

`use_sign` Whether to use a sign in the model for w, or just to weight all variants the same. Uses the sign of the estimated covariance between each marker and the trait to determine the sign of the marker.

`use_weight` Whether to use a weight in the model for w (inverse variance of allele frequency in everyone (continuous) or just the controls (dichotomous)).

`nperm` Number of permutations to run.

`binary` Is the phenotype dichotomous, or continuous? (TRUE/FALSE)

`strategy` ‘step’ uses the step-up routine (default, recommended), ‘afreq’ tests all allele frequencies, and ‘thresh’ uses a fixed threshold.

`thresh` Allele frequency for inclusion, used only when `strategy=`‘thresh’.

`genotype_gene` For `rarePathwayTest`, this is a vector of length `ncol(genotype)`. This can be strings to indicate what gene each variant is in; e.g. `c("brca1", "brca1", "brca1", "brca2", "brca2")` would indicate that the first three variants were in the first gene, and the last two variants are in the second gene. It can also be numeric.

`CUT` For `rarePathwayTest`, genes bigger than `CUT` will be chopped into subpieces of size smaller than `CUT` (for computational speed).

Details

The methods here are as described in the Hoffmann et al. (PLoS ONE, to appear) paper. These methods are based on the idea of trying multiple models for rare variants, since prior information is generally not very accurate. The p-value is corrected for multiple comparisons by permutation.

References

Hoffmann, TJ, Marini, NJ, and Witte, JS. Comprehensive approach to analyzing rare variants. PLoS ONE, 5(11): e13584. https://doi.org/10.1371/journal/pone.0013584.

Examples

```
## This inefficient (for clarity) code will simulate
##  a dataset in the correct format, and then run
##  the different approaches described here.

## Constants for data generation
SEED <- 2
NCASE <- NCONT <- 500
NGENE <- 6
FUNCTIONAL <- 1:(NGENE/2)  ## Half are functional
BETA0 <- -4
BETAG <- log(2)

set.seed(SEED) ## Reproducible results

nonfunctional <- setdiff(1:NGENE, FUNCTIONAL)
afreq <- runif(NGENE, 0.001, 0.01)

expit <- function(x)
  return(exp(x) / (1 + exp(x)))

gcase <- matrix(0, nrow=NCASE, ncol=NGENE)
for(indiv in 1:NCASE){
  affected <- FALSE
  while(!affected){ ## while not affected
    gcase[indiv, ] <- rbinom(NGENE, 2, afreq)  ## draw up genotype
    affected <- (expit(BETA0 +
      BETAG*sum(gcase[indiv, FUNCTIONAL])) > runif(1))
  }
}
cat("\n")

gcont <- matrix(0, nrow=NCONT, ncol=NGENE)
for(indiv in 1:NCONT){
  unaffected <- FALSE
  while(!unaffected){ ## while not unaffected
    gcont[indiv, ] <- rbinom(NGENE, 2, afreq) ## draw up genotype
    unaffected <- (1-expit(BETA0 +
      BETAG*sum(gcont[indiv, FUNCTIONAL])) > runif(1))
  }
}
cat("\n")

cat("# Rare functional variants cases =",
  sum(gcase[,FUNCTIONAL]), "\n")
cat("# Rare functional variants controls =",
  sum(gcont[,FUNCTIONAL]), "\n")
cat("# Rare non-functional variants cases =",
  sum(gcase[,nonfunctional]), "\n")
cat("# Rare non-functional variants controls =",
  sum(gcont[,nonfunctional]), "\n")
```

```
case <- c(rep(1,NCASE), rep(0,NCONT))
genotype <- rbind(gcase, gcont)

cat("P-value of the test:\n")
rareGeneTest(genotype, case)
```

Index