Package ‘hapassoc’

October 13, 2022

Version  1.2-9

Title    Inference of Trait Associations with SNP Haplotypes and Other
         Attributes using the EM Algorithm

Author   K. Burkett <kburkett@uottawa.ca>, B. McNeney <mcneney@sfu.ca>, J.
         Graham <jgraham@stat.sfu.ca>, with code for case-control data
         contributed by Zhijian Chen <z11chen@math.uwaterloo.ca>

Maintainer K. Burkett <kburkett@uottawa.ca>

Depends  R (>= 2.6.0), stats

Description The following R functions are used for inference of trait
         associations with haplotypes and other covariates in
         generalized linear models. The functions are developed
         primarily for data collected in cohort or cross-sectional
         studies. They can accommodate uncertain haplotype phase and
         handle missing genotypes at some SNPs.

License  GPL-2

URL      https://sfustatgen.github.io/research/hapassoc.html

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-05-25 20:10:02 UTC

R topics documented:

anova.hapassoc .............................................. 2
hapassoc ....................................................... 3
hypoDat .......................................................... 6
hypoDatGeno ..................................................... 6
logLik.hapassoc ............................................... 7
pre.hapassoc ................................................. 8
summary.hapassoc ........................................... 11

Index 13
anova.hapassoc

Description

This function returns the likelihood ratio test statistic comparing two nested models fit with hapassoc for cohort or cross-sectional data.

Usage

## S3 method for class 'hapassoc'
anova(object, redfit, display=TRUE, ...)

Arguments

object                a list of class hapassoc output by the hapassoc function.
redfit                A hapassoc object resulting from fitting a reduced model
display               An indicator to suppress output displayed on screen
...                   additional arguments to the summary function currently unused

Details

See the hapassoc vignette, of the same name as the package, for details.

Value

LRTstat           The likelihood ratio statistic comparing the two models
df                Degrees of freedom of the likelihood ratio statistic
pvalue            The p-value of the test

References


See Also

pre.hapassoc,hapassoc, summary.hapassoc.
**Examples**

```r
data(hypoDatGeno)
example2.pre.hapassoc <- pre.hapassoc(hypoDatGeno, numSNPs=3, allelic=FALSE)
example2.regr <- hapassoc(affected ~ attr + hAAA + hACA + hACC + hCAA + pooled, example2.pre.hapassoc, family=binomial())
example2.regr2 <- hapassoc(affected ~ attr + hAAA, example2.pre.hapassoc, family=binomial())
anova(example2.regr, example2.regr2)
```

# Returns:

# hapassoc: likelihood ratio test

#Full model: affected ~ attr + hAAA + hACA + hACC + hCAA + pooled
#Reduced model: affected ~ attr + hAAA
#LR statistic = 1.5433 , df = 4 , p-value = 0.8189

**Description**

This function takes a dataset of haplotypes in which rows for individuals of uncertain phase have been augmented by “pseudo-individuals” who carry the possible multilocus genotypes consistent with the single-locus phenotypes. For cohort or cross-sectional data, the EM algorithm is used to find MLE’s for trait associations with covariates in generalized linear models. For case-control data, the algorithm solves a set of unbiased estimating equations (see Details).

**Usage**

```r
hapassoc(form, haplos.list, baseline = "missing", family = binomial(), design="cohort", disease.prob=NULL, freq = NULL, maxit = 50, tol = 0.001, start = NULL, verbose=FALSE)
```

**Arguments**

- **form**: model equation in usual R format
- **haplos.list**: list of haplotype data from `pre.hapassoc`
- **baseline**: optional, haplotype to be used for baseline coding if the model formula either includes all haplotypes or is of the form "y~." for example. Default is the most frequent haplotype according to the initial haplotype frequency estimates returned by `pre.hapassoc`.
- **family**: binomial, poisson, gaussian or gamma are supported, default=binomial
study design. Default is "cohort" for cohort or cross-sectional sampling. Users may optionally specify "cc" for case-control or retrospective sampling of exposures (i.e. genotypes and non-genetic attributes) conditional on disease status. When design="cc", family=binomial() is assumed and the robust MPSE estimator of the regression parameters (Spinka et al., 2005) is returned; see Details for more information.

design study design. Default is "cohort" for cohort or cross-sectional sampling. Users may optionally specify "cc" for case-control or retrospective sampling of exposures (i.e. genotypes and non-genetic attributes) conditional on disease status. When design="cc", family=binomial() is assumed and the robust MPSE estimator of the regression parameters (Spinka et al., 2005) is returned; see Details for more information.

disease.prob marginal disease probability [P(D=1)] to use in the MPSE estimator, if design="cc". If disease.prob=NULL (the default value), a rare disease is assumed. This argument is ignored if design="cohort".

freq initial estimates of haplotype frequencies, default values are calculated in pre.hapassoc using standard haplotype-counting (i.e. EM algorithm without adjustment for non-haplotype covariates)

maxit maximum number of iterations of the EM algorithm; default=50

tol convergence tolerance in terms of either the maximum difference in parameter estimates between iterations or the maximum relative difference in parameter estimates between iterations, which ever is larger.

start starting values for parameter estimates in the risk model

verbose should the iteration number and value of the convergence criterion be printed at each iteration of the EM algorithm? Default=FALSE

Details

See the hapassoc vignette, of the same name as the package, for details.

When the study design is case-control, i.e. genotypes and non-genetic attributes have been sampled retrospectively given disease status, naive application of prospective maximum likelihood methods can yield biased inference (Spinka et al., 2005, Chen, 2006). Therefore, when design="cc", the algorithm solves the modified prospective score equations or MPSE (Spinka et al. 2005) for regression and haplotype frequency parameters. The implementation in hapassoc is due to Chen (2006). In general, the MPSE approach requires that the marginal probability of disease, P(D=1), be known. An exception is when the disease is rare; hence, when disease.prob=NULL (the default) a rare disease is assumed. The variance-covariance matrix of the regression parameter and haplotype frequency estimators is approximated as described in Chen (2006). Limited simulations indicate that the resulting standard errors for regression parameters perform well, but not the standard errors for haplotype frequencies, which should be ignored. For case-control data, we hope to implement the variance-covariance estimator of Spinka et al. (2005) in a future version of hapassoc.

Value

it number of iterations of the EM algorithm

beta estimated regression coefficients

freq estimated haplotype frequencies

fits fitted values of the trait

wts final weights calculated in last iteration of the EM algorithm. These are estimates of the conditional probabilities of each multilocus genotype given the observed single-locus genotypes.
hapassoc

var joint variance-covariance matrix of the estimated regression coefficients and the
estimated haplotype frequencies
dispersion maximum likelihood estimate of dispersion parameter (to get the moment esti-
mate, use \texttt{summary.hapassoc}) if applicable, otherwise 1
family family of the generalized linear model (e.g. binomial, gaussian, etc.)
response trait value
converged TRUE/FALSE indicator of convergence. If the algorithm fails to converge, only
the converged indicator is returned.
model model equation
loglik the log-likelihood evaluated at the maximum likelihood estimates of all param-
eters if \texttt{design}="cohort", or NA if \texttt{design}="cc"
call the function call

References

haplotypes and other attributes in generalized linear models. Human Heredity, \textbf{57}:200-206


Chen, Z. (2006): Approximate likelihood inference for haplotype risks in case-control studies of
a rare disease, Masters thesis, Statistics and Actuarial Science, Simon Fraser University, available at

environmental factors with missing genetic information and haplotype-phase ambiguity. Genetic
Epidemiology, \textbf{29}, 108-127.

See Also

\texttt{pre.hapassoc, summary.hapassoc, glm, family}.

Examples

data(hypoDat)
example.pre.hapassoc<-pre.hapassoc(hypoDat, 3)

example.pre.hapassoc$initFreq # look at initial haplotype frequencies
# h000  h001  h010  h011  h100  h101  h110  h111
#0.25179111 0.26050418 0.23606001 0.09164470 0.10133627 0.02636844 0.01081260
# h111
#0.02148268

names(example.pre.hapassoc$haploDM)
# "h000"  "h001"  "h010"  "h011"  "h100"  "h101"  "h110"  "pooled"

# Columns of the matrix haploDM score the number of copies of each haplotype
# for each pseudo-individual.

# Logistic regression for a multiplicative odds model having as the baseline
# group homozygotes '001/001' for the most common haplotype

example.regr <- hapassoc(affected ~ attr + h000 + h010 + h011 + h100 + pooled,
                          example.pre.hapassoc, family=binomial())

# Logistic regression with separate effects for 000 homozygotes, 001 homozygotes
# and 000/001 heterozygotes

example2.regr <- hapassoc(affected ~ attr + I(h000==2) + I(h001==2) +
                          I(h000==1 & h001==1), example.pre.hapassoc, family=binomial())

hypoDat

Simulated data for a hypothetical binary trait

**Description**

Simulated binary trait data used to illustrate the hapassoc package.

**Usage**

data(hypoDat)

**Format**

Matrix with columns:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[.1]</td>
<td>affected</td>
</tr>
<tr>
<td>[.3]</td>
<td>attr</td>
</tr>
<tr>
<td>[.5]</td>
<td>M1.1</td>
</tr>
<tr>
<td>[.6]</td>
<td>M1.2</td>
</tr>
<tr>
<td>[.5]</td>
<td>M2.1</td>
</tr>
<tr>
<td>[.6]</td>
<td>M2.2</td>
</tr>
<tr>
<td>[.7]</td>
<td>M3.1</td>
</tr>
<tr>
<td>[.8]</td>
<td>M3.2</td>
</tr>
</tbody>
</table>

hypoDatGeno

Simulated data for a hypothetical genetic SNPs

**Description**

Simulated genetic SNPs data used to illustrate the hapassoc package.
logLik.hapassoc

Usage

data(hypoDatGeno)

Format

Matrix with columns:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>affected</td>
<td>numeric</td>
<td>affection status (1=yes, 0=no)</td>
</tr>
<tr>
<td>[2]</td>
<td>attr</td>
<td>numeric</td>
<td>simulated quantitative attribute</td>
</tr>
<tr>
<td>[3]</td>
<td>M1</td>
<td>numeric</td>
<td>hypothetical SNP M1</td>
</tr>
</tbody>
</table>

logLik.hapassoc Return log-likelihood

Description

This function is used to return the log-likelihood at the maximum likelihood estimates computed by hapassoc and to return the number of parameters fit by hapassoc (i.e. the degrees of freedom in R) for cohort or cross-sectional data.

Usage

## S3 method for class 'hapassoc'
logLik(object, ...)

Arguments

object a list of class hapassoc output by the hapassoc function
... additional arguments to the summary function (currently unused)

Details

See the hapassoc vignette, of the same name as the package, for details.

Value

logLik log-likelihood computed at the maximum likelihood estimates if design="cohort", or NA if design="cc"
df number of parameters in the model (i.e. regression coefficients, any dispersion parameters and haplotype frequencies). This is not the residual degrees of freedom, which is the number of subjects minus the number of parameters estimated.
References

See Also
pre.hapassoc, hapassoc, summary.hapassoc.

Examples
```r
data(hypoDatGeno)
example2.pre.hapassoc<-pre.hapassoc(hypoDatGeno, numSNPs=3, allelic=FALSE)
example.regr <- hapassoc(affected ~ attr + hAAA + hACA + hACC + hCAA + pooled, example2.pre.hapassoc, family=binomial())
logLik(example.regr)
```

# Returns:
# Log Lik: -322.1558 (df=14)

---

pre.hapassoc

Pre-process the data before fitting it with hapassoc

Description
This function takes as an argument a dataframe with non-SNP and SNP data and converts the genotype data at single SNPs (the single-locus genotypes) into haplotype data. The rows of the input data frame should correspond to subjects. Single-locus SNP genotypes may be specified in one of two ways: (i) as pairs of columns, with one column for each allele of the single-locus genotypes (“allelic format”), or (ii) as columns of two-character genotypes (“genotypic format”). The SNP data should comprise the last 2*numSNPs columns (allelic format) or the last numSNPs columns (genotypic format) of the data frame.

If the haplotypes for a subject cannot be inferred from his or her genotype data, “pseudo-individuals” representing all possible haplotype combinations consistent with the single-locus genotypes are considered. Missing single-locus genotypes, up to a maximum of maxMissingGenos (see below), are allowed, but subjects with missing data in more than maxMissingGenos, or with missing non-SNP data, are removed. Initial estimates of haplotype frequencies are then obtained using the EM algorithm applied to the genotype data pooled over all subjects. Haplotypes with frequencies below a user-specified tolerance (zero.tol) are assumed not to exist and are removed from further consideration. (Pseudo-individuals having haplotypes of negligible frequency are deleted and the column in the design matrix corresponding to that haplotype is deleted.) For the remaining haplotypes, those with non-negligible frequency below a user-defined pooling tolerance (pooling.tol) are pooled into a single category called “pooled” in the design matrix for the risk model. However, the frequencies of each of these pooled haplotypes are still calculated separately.
Usage

pre.hapassoc(dat, numSNPs, maxMissingGenos=1, pooling.tol = 0.05,
zero.tol = 1/(2 * nrow(dat) * 10), allelic=TRUE, verbose=TRUE)

Arguments

- **dat**: the non-SNP and SNP data as a data frame. The SNP data should comprise the last 2* numSNPs columns (allelic format) or last numSNPs columns (genotypic format). Missing allelic data should be coded as NA or "" and missing genotypic data should be coded as, e.g., "A" if one allele is missing and "" if both alleles are missing.
- **numSNPs**: number of SNPs per haplotype
- **maxMissingGenos**: maximum number of single-locus genotypes with missing data to allow for each subject. (Subjects with more missing data, or with missing non-SNP data are removed.) The default is 1.
- **pooling.tol**: pooling tolerance – by default set to 0.05
- **zero.tol**: tolerance for haplotype frequencies below which haplotypes are assumed not to exist – by default set to $\frac{1}{2N+10}$ where N is the number of subjects
- **allelic**: TRUE if single-locus SNP genotypes are in allelic format and FALSE if in genotypic format; default is TRUE.
- **verbose**: indicates whether or not a list of the genotype variables used to form haplotypes and a list of other non-genetic variables should be printed; default is TRUE.

Details

See the hapassoc vignette, of the same name as the package, for details.

Value

- **haploTest**: logical, TRUE if some haplotypes had frequency less than zero.tol and are assumed not to exist
- **initFreq**: initial estimates of haplotype frequencies
- **zeroFreqHaplos**: list of haplotypes assumed not to exist
- **pooledHaplos**: list of haplotypes pooled into a single category in the design matrix
- **haploDM**: Haplotype portion of the data frame augmented with pseudo-individuals. Has 2^{numSNPs} columns scoring number of copies of each haplotype for each pseudo-individual
- **nonHaploDM**: non-haplotype portion of the data frame augmented with pseudo-individuals
- **haploMat**: matrix with 2 columns listing haplotype labels for each pseudo-individual
- **wt**: vector giving initial weights for each pseudo-individual for the EM algorithm
- **ID**: index for each individual in the original data frame. Note that all pseudo-individuals have the same ID value
References


See Also

hapassoc, summary.hapassoc.

Examples

# First example data set has single-locus genotypes in "allelic format"
data(hypoDat)
example.pre.hapassoc<-pre.hapassoc(hypoDat, numSNPs=3)

# To get the initial haplotype frequencies:
example.pre.hapassoc$initFreq
# h000  h001  h010  h011  h100  h101  h110
#0.25179111 0.26050418 0.23606001 0.09164470 0.10133627 0.02636844 0.01081260
# h111
#0.02148268
# The 'h001' haplotype is estimated to be the most frequent
example.pre.hapassoc$pooledHaplos
# "h101" "h110" "h111"
# These haplotypes are to be pooled in the design matrix for the risk model
names(example.pre.hapassoc$haploDM)
# "h000" "h001" "h010" "h011" "h100" "pooled"

####

# Second example data set has single-locus genotypes in "genotypic format"
data(hypoDatGeno)
example2.pre.hapassoc<-pre.hapassoc(hypoDatGeno, numSNPs=3, allelic=FALSE)

# To get the initial haplotype frequencies:
example2.pre.hapassoc$initFreq
# hAAA  hAAC  hACA  hACC  hCAA  hCAC
#0.25179111 0.26050418 0.23606001 0.09164470 0.10133627 0.02636844
# hCCA  hCCC
#0.01081260 0.02148268
# The 'hAAC' haplotype is estimated to be the most frequent
example2.pre.hapassoc$pooledHaplos
# "hCAC" "hCCA" "hCCC"
# These haplotypes are to be pooled in the design matrix for the risk model
names(example2.pre.hapassoc$haploDM)
# "hAAA" "hAAC" "hACA" "hACC" "hCAA" "pooled"
Summary function for reporting the results of the hapassoc function in a similar style to the lm and glm summaries.

### Usage

```r
## S3 method for class 'hapassoc'
summary(object, ...)
```

### Arguments

- `object` a list of class `hapassoc` output by the `hapassoc` function
- `...` additional arguments to the summary function (currently unused)

### Details

See the hapassoc vignette, of the same name as the package, for details.

### Value

- `call` The function call to hapassoc
- `subjects` The number of subjects used in the analysis
- `coefficients` Table of estimated coefficients, standard errors and Wald tests for each variable
- `frequencies` Table of estimated haplotype frequencies and standard errors
- `dispersion` Estimate of dispersion parameter (Moment estimator for gamma model)

### References


### See Also

`pre.hapassoc`, `hapassoc`
### Examples

```r
data(hypoDat)
exmple.pre.hapassoc <- pre.hapassoc(hypoDat, 3)
exmple.regr <- hapassoc(affected ~ attr + h000 + h010 + h011 + h100 + pooled,
                        example.pre.hapassoc, family=binomial())

# Summarize the results:
summary(example.regr)
```

|                | Estimate | Std. Error | zscore | Pr(>|z|) |
|----------------|----------|------------|--------|----------|
| (Intercept)    | -1.24114270 | 0.7820977 | -1.58694079 | 0.11252606 |
| attr           | 0.74036920  | 0.2918205 | 2.53707057  | 0.01117844  |
| h000           | 1.14968352  | 0.5942542 | 1.93466627  | 0.05303126  |
| h010           | -0.59318434 | 0.6569672 | -0.90291311 | 0.36657201  |
| h011           | -0.03615243 | 0.9161959 | -0.03945928 | 0.96852422  |
| h100           | -0.85329292 | 1.0203105 | -0.83630709 | 0.40298217  |
| pooled         | 0.38516864  | 0.8784283 | 0.43847478  | 0.66104215  |

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>f.h000</td>
<td>0.26716394</td>
<td>0.03933158</td>
</tr>
<tr>
<td>f.h001</td>
<td>0.25191674</td>
<td>0.03866739</td>
</tr>
<tr>
<td>f.h010</td>
<td>0.21997138</td>
<td>0.03881578</td>
</tr>
<tr>
<td>f.h011</td>
<td>0.10094795</td>
<td>0.02949617</td>
</tr>
<tr>
<td>f.h100</td>
<td>0.09507014</td>
<td>0.02371878</td>
</tr>
<tr>
<td>f.h101</td>
<td>0.02584918</td>
<td>0.01411881</td>
</tr>
<tr>
<td>f.h110</td>
<td>0.01779455</td>
<td>0.01386080</td>
</tr>
<tr>
<td>f.h111</td>
<td>0.02128613</td>
<td>0.01247265</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>dispersion</td>
<td>1</td>
</tr>
</tbody>
</table>
Index

* datasets
  hypoDat, 6
  hypoDatGeno, 6
* methods
  anova.hapassoc, 2
  hapassoc, 3
  logLik.hapassoc, 7
  pre.hapassoc, 8
  summary.hapassoc, 11
anova.hapassoc, 2
family, 5
glm, 5
hapassoc, 2, 3, 7, 8, 10, 11
hypoDat, 6
hypoDatGeno, 6
logLik.hapassoc, 7
pre.hapassoc, 2–5, 8, 8, 11
summary.hapassoc, 2, 5, 8, 10, 11