Package ‘HapEstXXR’

June 2, 2015

Version 0.1-8
Title Multi-Locus Stepwise Regression
Author Sven Knueppel and Klaus Rohde
Maintainer Sven Knueppel <sven.knueppel@dife.de>
Depends survival
Description The multi-locus stepwise regression (MSR) combines the advantages of stepwise regression and haplotype-based analysis. The MSR can be used to identify informative combinations of single nucleotide polymorphisms (SNPs) from unlinked SNPs (allele combinations) or SNPs within a chromosomal region (haplotypes).
License LGPL (>= 2.1)
Date 2015-06-02
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-06-02 16:36:52

R topics documented:

HapEstXXR-package ......................................................... 2
allele1to2 ................................................................. 2
catt ................................................................. 3
coding.baseline.allele ..................................................... 5
dec2bin ................................................................. 5
itegeppXXR ................................................................. 6
maf ................................................................. 7
makeHaplovviewInputFile .................................................. 8
makePlinkInputFile ......................................................... 9
msr ................................................................. 11
multi.snp.test .............................................................. 13
order.families .............................................................. 13
powerset .............................................................. 14
read.data .............................................................. 15
read.haplovview .......................................................... 16
single.haplotype.test .................................................... 18
single.snp.test ............................................................ 19
Description

Routines for multi-locus stepwise regression with SNP genotypes

Author(s)

Sven Knueppel and Klaus Rohde

allele1to2

Convert genotype matrix from two different types

Description

(not supported for x-linked markers)

Usage

allele1to2(geno, marker.label = NULL, miss.val = NA)
allele2to1(geno, marker.label = NULL, miss.val = NA)
alleleRto1(geno, marker.label = NULL, miss.val = NA)
alleleRto2(geno, marker.label = NULL, miss.val = NA)
allele1toR(geno, marker.label = NULL, miss.val = c(-1, NA))
allele2toR(geno, marker.label = NULL, miss.val = NA)

Arguments

geno (m,n)-genotype matrix
m=number of individuals
type 1 and R: n=number of snps
type 2: n=2*number of snps

marker.label Vector of labels for marker, If a marker name is "SNP", its columns will be "SNP.1" and "SNP.2"

miss.val Vector of specified missing values.

Details

3 different types of genotype matrices:
Type 1 : 1-column genotype matrix : minor allele count (0,1,2)
Type 2 : 2-column genotype matrix : each marker has a pair of two columns (1/1, 1/2, 2/2)
Type R : 1-column genotype matrix : code (1 = 1/1, 3 = 1/2, 2 = 2/2)
**Value**

converted genotype matrix

**Author(s)**

Sven Knueppel

**Examples**

```r
## [A] allele1to2
N <- 10
ns <- 4
(geno <- matrix(sample(c(NA, 0:2), N * ns, replace = TRUE), nc = ns))
allele1to2(geno)

## [B] allele2to1
(geno <- matrix(c(0, 0, 1, 1, 2, 1, 2, 1, 1, 2, 2, 2, 1, 2, 0, 0, 1, 1, 2, 1, 0, 0), nc = 4, byrow = TRUE))
allele2to1(geno)

## [C] alleleRto1
N <- 10
ns <- 4
(geno <- matrix(sample(c(NA, 1:3), N * ns, replace = TRUE), nc = ns))
alleleRto1(geno)

## [D] alleleRto2
N <- 10
ns <- 4
(geno <- matrix(sample(c(0, 1:3), N * ns, replace = TRUE), nc = ns))
alleleRto2(geno)

## [E] allele1toR
N <- 10
ns <- 4
(geno <- matrix(sample(c(NA, 0:2), N * ns, replace = TRUE), nc = ns))
allele1toR(geno)

## [F] allele2toR
(geno <- matrix(c(0, 0, 1, 1, 2, 1, 2, 1, 1, 2, 2, 2, 1, 2, 0, 0, 1, 1, 2, 1, 0, 0),
    nc = 4, byrow = TRUE))
allele2toR(geno)
```

---

catt

Cochrane armitage trend test (CATT) for SNP genotypes
Description

Performs chi-squared test for SNP genotypes. By default, score is chosen as the number of alleles (0, 1, 2).

Usage

catt(y, x, score = c(0, 1, 2))

Arguments

y Vector of trait values. y must have values of 1 for event, 0 for no event.
x Vector of SNP genotypes, 1-column coding (SNP allele dosage: 0,1,2).
score Group score.

Details

The Cochran-Armitage trend test is typically used in categorical data analysis when some categories are ordered. Here it is used as a genotype-based test for candidate gene association.

Value

2x3-table Genotype distribution.
chisq The value for the test statistic.
df Degrees of freedom.
p.value The p-value for the test.
n.miss Number of individuals with missing values.

Author(s)

Sven Knueppel

References


See Also

prop.trend.test

Examples

y <- sample(c(0, 1), 100, replace = TRUE)
x <- sample(c(0, 1, 2), 100, replace = TRUE)
catt(y, x)
**coding.baseline.allele**

*Standardization of coding alleles*

**Description**

Dependend on minor allele frequency the coding of the alleles will be updated.

**Usage**

```r
coding.baseline.allele(geno, coding = c("minor", "major"))
```

**Arguments**

- `geno` 
  - (m,n)-genotype matrix
  - `m=number of individuals`
  - `type R: n=number of snps`
- `coding` 
  - which type of coding should be used.

**Details**

Allele 1 is coded as the minor allele, if coding type "minor" is used. Otherwise allele 1 is coded as major allele.

**Value**

This function returns the updated genotype matrix.

**Author(s)**

Sven Knueppel

---

**dec2bin**

*Decimal To Binary Conversion*

**Description**

`dec2bin` function converts a decimal number to a binary number.

**Usage**

```r
dec2bin(vec, npos = NA)
```
Arguments

vec  a numeric vector of positive values.
npos an optional number of length of the generating binary number.

Details

This is a function to converting from decimal to binary number.

Value

dec2bin returns a matrix.

Examples

```r
binary <- dec2bin(zz <- sample(0:100, 10))
print(zz)
print(binary)
```

itegeppXXR  

Haplotype estimation routine for single individual data

Description

itegeppXXR is haplotype estimation routine for samples of independent individual genotypes (EM-algorithm).

Usage

```r
itegeppXXR(geno, des = 0, lim = 0.05)
```

Arguments

geno  (n,m)-Matrix; n=No of Individuals, m=No of SNPs; R-Code: 1-column genotype matrix - code 1 = 1/1, 3 = 1/2, 2 = 2/2
des  des=1 haplotype pairs, des=0 single haplotypes
lim  Threshold for combining rare haplotypes

Details

Inferring haplotypes by EM-Algorithm
### maf

**Value**

- **hap.id**  
  Id. of haplotypes
- **hap**  
  estimated haplotypes
- **freq**  
  haplotype frequencies
- **hapres**  
  individual haplotypes
- **likres**  
  Likelihood value
- **desres**  
  Design matrix for the model (des=1 => Haplotype pairs, des=0 => single haplotypes)

**Note**

This function works only up to 15 SNP haplotypes

**Author(s)**

Sven Knueppel and Klaus Rohde

**References**


**Examples**

```r
set.seed(123456)
ns <- 4  # Number of SNPs
N <- 2000  # Number of individuals
patid <- N:1
geno <- matrix(sample(c(1, 2, 3), ns * N, replace = TRUE), ncol = ns)
iteHAP <- itegeppXXR(geno, des = 1, lim = 0.01)
```

<table>
<thead>
<tr>
<th>maf</th>
<th>Minor alle frequencies</th>
</tr>
</thead>
</table>

**Description**

Calculation of minor allele frequencies (MAF), call rate and asymptotic chi-square hardy-weinberg test

**Usage**

```r
maf(geno, marker.label = NA)
```
makeHaploviewInputFile

Arguments

- geno: (m, n)-genotype matrix
  m = number of individuals
  type R: n = number of snps

- marker.label: Labels for the markers.

Details

Call rate is defined by number of missing genotypes divided by sample size.
Testing deviation of the hardy-weinberg equilibrium is done by the usual goodness-of-fit chisquare
test: \( \text{chi2} \sim \sum (\text{observed-expected})^2 / \text{expected} \).

Value

This function returns a matrix with 8 columns.

Author(s)

Sven Knueppel

makeHaploviewInputFile

Make Haploview input files

Description

Create two data sets (*.ped and *.info) as input files for Haploview

Usage

makeHaploviewInputFile(famid, patid, fid, mid, sex,
aff, geno, marker.name, marker.position,
haploview.pedfile, haploview.infofile)

Arguments

- famid: Family ID
- patid: Individual ID
- fid: Paternal ID
- mid: Maternal ID
- sex: 1=male, 2=female, other=unknown
- aff: disease phenotype (1=unaff, 2=aff, 0=missing/unknown)
- geno: (n,m) genotype matrix (n=number of individuals, m=number of marker, 1-column for every marker, R-code: 1 = 1/1, 3 = 1/2, 2 = 2/2); All markers should be biallelic.
makePlinkInputFile

makePlinkInputFile('
marker.name marker name
marker.position marker position
haploview.pedfile specify target of linkage file
haploview.infofile specify target of marker Information file

Details
This function provides only limited options for creating Haploview input files. For more details see Haploview/URL: http://www.broadinstitute.org/mpg/haploview.

Value
no return values.

Author(s)
Sven Knueppel

References
Haploview/URL: http://www.broadinstitute.org/mpg/haploview

See Also
makePlinkInputFile, allele1to2

makePlinkInputFile Make PLINK input files

Description
Create two data sets (*.ped and *.map) as input files for PLINK

Usage
makePlinkInputFile(famid, patid, fid, mid, sex, trait, CHR, SNP, POS, geno.matrix, linkage.file, map.file, cov.file)
Arguments

- **famid**: Family ID
- **patid**: Individual ID
- **fid**: Paternal ID
- **mid**: Maternal ID
- **sex**: 1=male, 2=female, other=unknown
- **trait**: Disease phenotype (1=unaff, 2=aff, -9 or 0=missing/unknown)
- **CHR**: Chromosome
- **SNP**: Marker name
- **POS**: Marker position
- **geno.matrix**: (n,m) genotype matrix (n=number of individuals, m=number of marker, 1-column for every marker, R-code: 1 = 1/1, 3 = 1/2, 2 = 2/2); All markers should be biallelic.
- **linkage.file**: Specify target of linkage file
- **map.file**: Specify target of map file
- **cov.file**: Specify target of cov file

Details

This function provides only limited options for creating PLINK input files. For more details see PLINK/URL: [http://pngu.mgh.harvard.edu/~purcell/plink/](http://pngu.mgh.harvard.edu/~purcell/plink/).

Value

No return values.

Author(s)

Sven Knueppel

References

Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MAR, Bender D, Maller J, Sklar P, de Bakker PIW, Daly MJ & Sham PC (2007) PLINK: a toolset for whole-genome association and population-based linkage analysis. American Journal of Human Genetics, 81. (PLINK/URL: [http://pngu.mgh.harvard.edu/~purcell/plink/](http://pngu.mgh.harvard.edu/~purcell/plink/))

See Also

- [makeHaploviewInputFile](#)
- [allele1toR](#)
Multi-locus stepwise regression

Description

Stepwise regression for SNP selection and haplotype testing

Usage

msr(snps, trait, famid, patid, fid, mid,
    adj.var = NA, lim = 0.05, maxSNP = 3,
    nt = 10, sort.by = "AICc", selection = 0,
    p.threshold = NA,
    pair.begin = FALSE, pattern.begin.mat = NA,
    type = "gaussian",
    baseline.hap = "max", min.count = 10, sort = FALSE)

Arguments

- snps: (n, m)-Matrix; n=No. of individuals; m=no. of SNPs; Rohde-Code
- trait: numeric; Outcome, phenotype
- famid: vector; Identifier for every family; only need in case of type=families
- patid: vector; Identifier for every individuals; only need in case of type=families
- fid: vector; Identifier for father (0=unknown); only need in case of type=families
- mid: vector; Identifier for mother (0=unknown); only need in case of type=families
- adj.var: (n, m)-Matrix; n = No. of individuals; m = no. of covariates; variables for adjustment
- lim: numeric; threshold for skipping haplotypes from analysis
- maxSNP: integer; Number of SNPs maximal group to multilocus genotypes
- nt: integer; Number of notice best hits (for every step)
- sort.by: the results in each step were sorted by "AIC", corrected ("AICc"), or p value ("p.value"). default = "AICc".
- selection: 0 = none, 1 = improve of the lowest corrected AIC (AICc) of the step before, 2 = improve of the lowest AIC of the step before, 3 = improve of p value, 4 = improve of best ten log10(p values), 5 = improve of the single AICc by adding one SNP to the noticed pattern
- p.threshold: numeric vector; if global p value is lower than p.threshold[i], then the pattern will be stored for further processing. i indicates the number of SNPs. If your calculation should start with all pairwise SNPs, then p.threshold[1] will be not used but should be included.
- pair.begin: If true then will be begin with first 2 SNP genotypes. Attention: k SNP lead to choose(k, 2) = k * (k - 1) / 2 possible pairs
if begin.pattern.mat is not NA then is this starting point of msr

n = No. of snp
m = No. of SNPs

type

type of depending variable

baseline.hap

Choose baseline haplotype for statistical test to avoid singularity. "max" for most frequent haplotype and "min" for less frequent haplotype

min.count

minimal count of rare haplotypes. If the count of estimated haplotypes < min.count, then the combined rare haplotypes were excluded from the analysis of that specific pattern.

sort

A logical value (TRUE or FALSE). If TRUE, family data will be sorted.

Details

Haplotypes are inferred by EM algorithm (Excoffier and Slatkin 1995). Family haplotypes are inferred by modified EM algorithm proposed by Rohde (2001, 2003).

For normal distributed phenotypes from independent individuals we prefer an F test and for case control data we prefer the likelihood ratio test (logistic regression) in comparison of full model with genetic and non-genetic factors to a reduced model, which includes only non-genetic variables. In the case of no specified non-genetic variable only the intercept is used. If one of these tests is significant we assume a genetic effect. In case of family data the weighted TDT statistic is used.

The procedure of multi-locus stepwise regression could be time consuming.

Value

msr provides a list with maxSNP components.

list for every step one component: SNP numbers and test details.

Author(s)

Sven Knueppel and Klaus Rohde

References

multi.snp.test

**Internal function used for multi-locus associations tests.**

**Description**

This function is used for internal computations. You should not use it, but you could.

**Usage**

```r
multi.snp.test(y, x, x.adj = NULL,
               type = c("gaussian", "binomial"))
```

**Arguments**

- `y`: response
- `x`: Matrix including SNPs or haplotypes
- `x.adj`: Matrix of covariates
- `type`: type of response

**Author(s)**

Sven Knueppel

---

**order.families**

**Ordering of nuclear family data**

**Description**

`order.families` returns a permutation which rearranges the families into ascending famid, generation, and sex, if given.

**Usage**

```r
order.families(famid, patid, fid, mid, sex = NA)
```

**Arguments**

- `famid`: vector; Identifier for every family
- `patid`: vector; Identifier for every individual
- `fid`: vector; Identifier for father (0 = unknown)
- `mid`: vector; Identifier for mother (0 = unknown)
- `sex`: vector; Individuals’ gender (1 = male, 2 = female, 0 = unknown)
Author(s)

Sven Knuepapel

Examples

fam <- as.character(c(c(1, 1, 1), c(0, 0, 0, 0, 0)))
pid <- as.character(c(c(1, 2, 3, 4), c(7, 8, 9, 10, 11)))
mid <- as.character(c(c(3, 3, 0, 0), c(10, 10, 0, 0)))
fid <- as.character(c(c(4, 4, 0, 0), c(11, 11, 11, 0, 0)))
sex <- as.character(c(c(0, 2, 2), c(1, 1, 2, 1)))
ordfam <- order.families(fam, pid, fid, mid, sex)
print(cbind(fam, pid, fid, mid, sex)[ordfam, ])

powerset

Generating power set of a set

Description

Generates the power set of a given set of values.

Usage

powerset(x, fileout = NA, only.file = FALSE)

Arguments

x

a vector.

fileout

a character string which contains name of the target file.

only.file

a logical. Only a file is created, if true (default=FALSE).

Details

Suppose you have a set S. The power set is the set of all subsets of S, including empty set and S itself. The number of elements of the power set is \(2^n\) (number of elements of S). You can save the powerset in a file, if a filename fileout is specified.

Empty set will be excluded.

Value

powerset generates a list of all subsets of x, excluding empty set, if only.file=F.
Note
powerset is restricted to vectors with maximum number of 15 elements. Using only file=TRUE you can create bigger powersets.

Author(s)
Sven Knueppel

Examples
```r
ps <- powerset(1:10)
ps
```

---

**read.data**

*Read data from different input files*

Description
Data can be loaded in different formats.

Usage
```r
read.data(filename, linkage = TRUE, map = NA)
```

Arguments
- `filename` the name of the file which the data are to be read from.
- `linkage` a logical value indicating whether the file is in linkage format.
- `map` Localization of a map file.

Details
1) single individuals (3-columns)
   - expected columns
     - Individual identifier
     - genotype STRING (1=homozygot (wildtype) 2=homozygot (variant) 3=heterozygote 0=missing value) » Example: "1223" "3023"
     - phenotype

2) family data (4-columns)
   - expected columns
     - Family identifier
     - Individual identifier
     - genotype (1=homozygot (wildtype) 2=homozygot (variant) 3=heterozygote 0=missing value) » Example: "1223" "3023"
phenotype
Remark 1: patid should not be 0 because 0 is unkown value for fid and mid.
Remark 2: Families are sorted. First two person in a family are adults (father and then mother) and after that all children.

3) Linkage format is expected, if linkage=TRUE :
Family identifier
Individual identifier
Father identifier (0=unkown)
Mother identifier (0=unkown)
Sex (0=unkown,1=maile,2=female)
Affectation_status (0=unkown,1=unaffected,2=affected) or trait_value
Marker_genotypes (M1_A1 M1_A2 M2_A1 ...) » only 1, 2, or 0 for missing values

4) map file (4-columns), if specified:
chromosome (1-22, only autosomes)
snp identifier
Genetic distance (morgans)
Base-pair position (bp units)

Value

- famid: family identifier
- patid: individual identifier
- fid: father identifier (0=unkown)
- mid: mother identifier (0=unkown)
- sex: sex (0=unkown,1=male,2=female)
- genotypes: (n,m)-matrix; n=No. of individuals; m=No. of SNPs; Klaus format
- trait: phenotype values
- chr: chromosome
- snp: snp identifier or rs id
- pos: Base-pair position on chromosome (base pair units)

Author(s)
Sven Knueppel

---

**read.haploview**

**Read a haploview dataset**

**Description**

Data can be loaded in haploview format (linkage format) with columns of family, individual, father, mother, gender (1 = male, 2 = male), affected status (0 = unkown, 1 = unaffected, 2 = affected), and genotypes(2 columns alleles).
Usage

read.haploview(ped.file, map.file)

Arguments

ped.file Localizion of a pedigree file.
map.file Localizion of a marker information file.

Details

The marker information file should contain in the first column the marker name and the second column the physical position on the chromosome.

Value

famid family identifier
patid individual identifier
dad father identifier (0=unknown)
mom mother identifier (0=unknown)
sex sex (0=unknown, 1=male, 2=female)
genotypes (n,m)-matrix; n=No. of individuals; m=No. of SNPs; 1-column allele dosis
trait phenotype values
marker.names marker.names
marker.position Base-pair position on chromosome (base pair units)

Author(s)

Sven Knueppel

References

Haploview/URL: http://www.broadinstitute.org/mpg/haploview

See Also

read.data
single.haplotype.test  single haplotype test

Description

Association test based on haplotypes. Haplotypes are estimated by EM algorithm.

Usage

```
single.haplotype.test(snps, trait, famid, patid, fid, mid,
    adj.var = NULL, type = c("gaussian", "binomial", "families"),
   prt = TRUE, lim = 0.05, min.count = 10,
    alpha = 0.05, sort = FALSE)
```

Arguments

- `snps`  (n.m)-Matrix; n=No. of individuals; m=no. of SNPs; Rohde-Code
- `trait` numeric; Outcome, phenotype
- `famid` vector; Identifier for every family; only need in case of type=families
- `patid` vector; Identifier for every individuals; only need in case of type=families
- `fid` vector; Identifier for father (0=unkown); only need in case of type=families
- `mid` vector; Identifier for mother (0=unkown); only need in case of type=families
- `adj.var`  (n,m)-Matrix; n=No. of individuals; m=no. of covariates; variables for adjustment; in case of type=families not available.
- `type`  type of depending variable
- `lim` numeric; threshold for pooling of haplotypes and declare as rare.
- `min.count` Minimal count for using pooled rare haplotypes in the analysis.
- `prt` A logical value (TRUE or FALSE). If TRUE, an overview is printed.
- `alpha` In case of type=binomial the (1-alpha/2)-confidence intervals are computed.
- `sort` A logical value (TRUE or FALSE). Only usable with family data. If TRUE, families are sorted by famid and generation which is a condition of wTDT.

Details

Haplotypes are infered by EM algorithm (Excoffier and Slatkin 1995).
For normal distributed phenotypes from independent individuals we prefer an F test and for case control data we prefer the likelihood ratio test (logistic regression) in comparison of full model with genetic and non-genetic factors to a reduced model, which includes only non-genetic variables. In the case of no specified non-genetic variable only the intercept is used. If one of these tests are significance we assume a genetic effect. In case of family data the weighed TDT statistic is used.
Value

- hap: Haplotypes
- freq: Estimated haplotype frequencies
- global.test: Result of global test statistic.
- haplotype.i: Result of haplotype specific tests

Author(s)

Sven Knueppel and Klaus Rohde

References


See Also

- single.snp.test

---

**single.snp.test**

Regression analysis with single SNP genotypes as independent variable

---

**Description**

This function fits a generalized linear model with quantitative, dichotomous or survival trait as dependent variable and one or more potential covariates. In case of family data the weighted TDT statistic is used.

**Usage**

```r
single.snp.test(snps, trait, adj.var = NULL,
    type = c("gaussian", "binomial", "families", "casecohort"),
    famid, patid, fid, mid,
    start.time, stop.time, subcohort, stratvar = NA, robust = FALSE,
    marker.label = NA,
    prt = TRUE, ties = "efron")
```
Arguments

- **snps**: Matrix of alleles, such that each locus has one column of alleles (R code: 1 = 1/1, 3 = 1/2, 2 = 2/2, 0 = missing). Rows contains alleles for each subject.
- **trait**: Vector of trait values. For case control data use type= "binomial", trait must have values of 1 for event, 0 for no event.
- **adj.var**: Matrix of (non-genetic) covariates used to adjust the regression model.
- **type**: Character string defining type of trait, with values of gaussian, binomial, families, survival, and casecohort.
- **famid**: vector; Identifier for every family; needed by type="families".
- **patid**: vector; Identifier for every individual; needed by type="families" and type="casecohort".
- **fid**: vector; Identifier for father (0=unknown); needed by type="families".
- **mid**: vector; Identifier for mother (0=unknown); needed by type="families".
- **start.time**: vector; age at the start of the follow-up.
- **stop.time**: vector; age at the end of the follow-up.
- **subcohort**: A logical value (TRUE or FALSE). If TRUE, the individual is in the subcohort.
- **stratvar**: vector; names the variables that determine the stratification.ss
- **robust**: A logical value (TRUE or FALSE). If TRUE, request the robust sandwich estimate.
- **marker.label**: Vector of labels for marker.
- **prt**: A logical value (TRUE or FALSE). If TRUE, an overview is printed.
- **ties**: defines the handling of ties in case-cohort design: "efron" (default),"breslow","exact".

Details

For normal distributed phenotypes from independent individuals we prefer an F test and for case control data we prefer the likelihood ratio test (logistic regression) in comparison of full model with genetic and non-genetic factors to a reduced model, which includes only non-genetic variables. In the case of no specified non-genetic variable only the intercept is used. If one of these tests are significant we assume a genetic effect. In case of family data the weighted TDT statistic is used.

So far SURVIVAL data is not supported.

Cox proportional hazards regression modified for case cohort designs according to the Prentice method will be used by type="casecohort".

As genetic effect the allele dosis (0, 1, 2) is modelled.

Value

single.snp.test returns an object of class data.frame containing the following components:

- **snp**: snp number
- **N**: number of individuals
- **type**: type of depending variable
- **beta**: estimation of beta coefficient out of full regression model
se(beta) estimation of standard error of beta coefficient out of full regression model
exp(beta) Odds ratio=exp(beta.estimate) are calculated, if type = "binomial". In case of type = "casecohort" hazard ratio is calculated.
lower.95 lower limit of 95 % confidence interval for exp(beta).
upper.95 upper limit of 95 % confidence interval for exp(beta).
aic Akaike’s An Information Criterion (AIC) of full model

Author(s)
Sven Knueppel and Klaus Rohde

References
Multi-locus stepwise regression: a haplotype-based algorithm for finding genetic associations applied to atopic dermatitis.

See Also
single.haplotype.test

Examples
N <- 2000
nloci <- 14
set.seed(1234)
y <- sample(c(0, 1), N, replace = TRUE)
snp <- matrix(sample(c(1, 2, 3), N * nloci, replace = TRUE),
ncol = nloci)
colnames(snp) <- paste("SNP", 1:nloci, sep = "")
adj.var <- matrix(rnorm(N * 3), ncol = 3)
colnames(adj.var) <- paste("A", 1:3, sep = "")
sst <- single.snp.test(snps = snp, trait = y, adj.var = adj.var,
type = "binomial",prt = TRUE)
Index

*Topic **SNPs**
  catt, 3
*Topic **SNP**
  catt, 3
*Topic **armitage**
  catt, 3
*Topic **binary**
  dec2bin, 5
*Topic **catt**
  catt, 3
*Topic **cochrane**
  catt, 3
*Topic **conversion**
  dec2bin, 5
*Topic **convert**
  allele1to2, 2
*Topic **decimal**
  dec2bin, 5
*Topic **families**
  order.families, 13
*Topic **haplotypes**
  single.haplotype.test, 18
  single.snp.test, 19
*Topic **haploview**
  read.haploview, 16
*Topic **input files**
  read.data, 15
*Topic **logistic regression**
  single.haplotype.test, 18
  single.snp.test, 19
*Topic **power set**
  powerset, 14
*Topic **powerset**
  powerset, 14
*Topic **read data**
  read.data, 15
*Topic **read.data**
  read.haploview, 16
*Topic **snp**
  single.haplotype.test, 18
  single.snp.test, 19
*Topic **sort**
  order.families, 13
*Topic **stepwise regression**
  msr, 11
*Topic **subset**
  powerset, 14
*Topic **survival analysis**
  single.haplotype.test, 18
  single.snp.test, 19
*Topic **test**
  catt, 3
*Topic **trend**
  catt, 3
*Topic **weighted TDT**
  single.haplotype.test, 18
  single.snp.test, 19
allele1to2, 2, 9, 10
allele1toR(allele1to2), 2
allele2to1(allele1to2), 2
allele2toR(allele1to2), 2
alleleRto1(allele1to2), 2
alleleRto2(allele1to2), 2
catt, 3
coding.baseline.allele, 5
dec2bin, 5
hap.est.caco(single.haplotype.test), 18
hap.est.gaussian
  (single.haplotype.test), 18
HapEstXXR(HapEstXXR-package), 2
HapEstXXR-package, 2
itegeppXXR, 6
maf, 7
makeHaploviewInputFile, 8, 10
INDEX

makePlinkInputFile, 9, 9
msr, 11
multi.snp.test, 13

order.families, 13

powerset, 14
prop.trend.test, 4

read.data, 15, 17
read.haploview, 16

single.haplotype.test, 18, 21
single.snp.test, 19, 19