Package ‘gap’

January 24, 2018

Version 1.1-21
Date 2018-1-23
Title Genetic Analysis Package
Author Jing Hua Zhao and colleagues with inputs from Kurt Hornik and Brian Ripley
Maintainer Jing Hua Zhao <jinghua.zhao@mrc-epid.cam.ac.uk>
Depends R (>= 2.10)
Suggests BradleyTerry2, MASS, Matrix, MCMCglmm, R2jags, bdsmatrix, coda, coxme, foreign, grid, haplo.stats, kinship2, lattice, magic, mets, nlme, pedigree, pedigreeMM, regress, rms, survival
LazyData Yes
LazyLoad Yes
Description It is designed as an integrated package for genetic data analysis of both population and family data. Currently, it contains functions for sample size calculations of both population-based and family-based designs, probability of familial disease aggregation, kinship calculation, statistics in linkage analysis, and association analysis involving genetic markers including haplotype analysis with or without environmental covariates.
License GPL (>= 2)
URL https://jinghuazhao.github.io
NeedsCompilation yes
Repository CRAN
Date/Publication 2018-01-24 11:40:17 UTC

R topics documented:

gap-package ................................................................. 3
ab ................................................................. 6
AE3 ................................................................. 7
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>aldh2</td>
</tr>
<tr>
<td>apoeapoc</td>
</tr>
<tr>
<td>asplot</td>
</tr>
<tr>
<td>b2r</td>
</tr>
<tr>
<td>BFDP</td>
</tr>
<tr>
<td>bt</td>
</tr>
<tr>
<td>ccsize</td>
</tr>
<tr>
<td>cf</td>
</tr>
<tr>
<td>chow.test</td>
</tr>
<tr>
<td>comp.score</td>
</tr>
<tr>
<td>crohn</td>
</tr>
<tr>
<td>ESplot</td>
</tr>
<tr>
<td>fa</td>
</tr>
<tr>
<td>fbsize</td>
</tr>
<tr>
<td>FPRP</td>
</tr>
<tr>
<td>fsnps</td>
</tr>
<tr>
<td>gc.em</td>
</tr>
<tr>
<td>gcontrol</td>
</tr>
<tr>
<td>gcontrol2</td>
</tr>
<tr>
<td>gcp</td>
</tr>
<tr>
<td>geneCounting</td>
</tr>
<tr>
<td>gif</td>
</tr>
<tr>
<td>h2</td>
</tr>
<tr>
<td>hap</td>
</tr>
<tr>
<td>hap.em</td>
</tr>
<tr>
<td>hap.score</td>
</tr>
<tr>
<td>hla</td>
</tr>
<tr>
<td>htr</td>
</tr>
<tr>
<td>hwe</td>
</tr>
<tr>
<td>hwe.cc</td>
</tr>
<tr>
<td>hwe.hardy</td>
</tr>
<tr>
<td>kin.morgan</td>
</tr>
<tr>
<td>klem</td>
</tr>
<tr>
<td>l51</td>
</tr>
<tr>
<td>LD22</td>
</tr>
<tr>
<td>LDk1</td>
</tr>
<tr>
<td>lukas</td>
</tr>
<tr>
<td>makeped</td>
</tr>
<tr>
<td>mao</td>
</tr>
<tr>
<td>masize</td>
</tr>
<tr>
<td>MCMCgrm</td>
</tr>
<tr>
<td>metap</td>
</tr>
<tr>
<td>metareg</td>
</tr>
<tr>
<td>meyer</td>
</tr>
<tr>
<td>mfblong</td>
</tr>
<tr>
<td>mhtplot</td>
</tr>
<tr>
<td>mhtplot2</td>
</tr>
<tr>
<td>mia</td>
</tr>
</tbody>
</table>
It is designed as an integrated package for genetic data analysis of both population and family data. Currently, it contains functions for sample size calculations of both population-based and family-based designs, classic twin models, probability of familial disease aggregation, kinship calculation, some statistics in linkage analysis, and association analysis involving one or more genetic markers including haplotype analysis with or without environmental covariates.

Details

Package: gap
Version: 1.1-20
Depends: R(>= 2.1.0)
Suggests: BradleyTerry2, MASS, Matrix, MCMCglmm, NCBI2R, bdsmatrix, coxme, foreign, grid, haplo.stats, kinship2, magic, mets, nlmixr, pedigree, pedigreemm, regress, rms, survival
License: GPL (>=2)
URL: http://www.mrc-epid.cam.ac.uk/people/jing-hua-zhao
* ANALYSIS *

AE3 AE model using nuclear family trios
bt Bradley-Terry model for contingency table
ccsize Power and sample size for case-cohort design
fbsize Sample size for family-based linkage and association design
gc.em Gene counting for haplotype analysis
gcontrol genomic control
gcontrol2 genomic control based on p values
gcp Permutation tests using GENECOUNTING
genecounting Gene counting for haplotype analysis
gif Kinship coefficient and genetic index of familiality
hap Haplotype reconstruction
hap.em Gene counting for haplotype analysis
hap.score Score statistics for association of traits with haplotypes
htr Haplotype trend regression
hwe Hardy-Weinberg equilibrium test for a multiallelic marker
hwe.cc A likelihood ratio test of population Hardy-Weinberg equilibrium
hwe.hardy Hardy-Weinberg equilibrium test using MCMC
kin.morgan kinship matrix for simple pedigree
LD22 LD statistics for two diallelic markers
LDkl LD statistics for two multiallelic markers
masize Sample size calculation for mediation analysis
MCMCgrm Mixed modeling with genetic relationship matrices
mia multiple imputation analysis for hap
mtdt Transmission/disequilibrium test of a multiallelic marker
mtdt2 Transmission/disequilibrium test of a multiallelic marker by Bradley-Terry model
mvmeta Multivariate meta-analysis based on generalized least squares
pbsize Power for population-based association design
pbsize2 Power for case-control association design
pfc Probability of familial clustering of disease
pfc.sim Probability of familial clustering of disease
pgc Preparing weight for GENECOUNTING
print.hap.score Print a hap.score object
s2k Statistics for 2 by K table
tscc Power calculation for two-stage case-control design

* GRAPHICS *

asplot Regional association plot
ESplot Effect-size plot
mhtplot Manhattan plot
mhtplot2 Manhattan plot with annotations
pedodot Converting pedigree(s) to dot file(s)
plot.hap.score Plot haplotype frequencies versus haplotype score statistics
qqfun Quantile-comparison plots
qqunif Q-Q plot for uniformly distributed random variable

* DATASETS *

PD A study of Parkinson’s disease and APOE, LRRK2, SNCA makers
aldh2 ALDH2 markers and alcoholism
apoapoc APOE/APOC1 markers and schizophrenia
cf Cystic Fibrosis data
crohn Crohn’s disease data
fa Friedreich ataxia data
fsnps A case-control data involving four SNPs with missing genotype
hla HLA markers and schizophrenia
l51 An example pedigree data
lukas An example pedigree
mao A study of Parkinson’s disease and MAO gene
meyer A pedigree data on 282 animals deriving from two generations
nep499 A study of Alzheimer’s disease with eight SNPs and APOE

* UTILITIES *

SNP Functions for single nucleotide polymorphisms (SNPs)
BFDP Bayesian false-discovery probability
FPRP False-positive report probability
ab Test/Power calculation for mediating effect
b2r Obtain correlation coefficients and their variance-covariances
chow.test Chow’s test for heterogeneity in two regressions
comp.score score statistics for testing genetic linkage of quantitative trait
h2 Heritability estimation according to twin correlations for case-control studies
klem Haplotype frequency estimation based on a genotype table of two multiallelic markers
makeped A function to prepare pedigrees in post-MAKEPED format
metap Meta-analysis of p values
metareg Fixed and random effects model for meta-analysis
muvar Means and variances under 1- and 2- locus (diallelic) QTL model
read.ms.output A utility function to read ms output
twinar90 Classic twin models
wshscore Whittemore-Halpern scores for allele-sharing
GRM functions ReadGRM, ReadGRMBin, ReadGRMPLINK, ReadGRMPCA, WriteGRM, WriteGRMBin, WriteGRMPLINK handle genomic relationship matrix involving other software
heritability functions h2G, VR, h2GC, h2I give point estimates as with their variances for continuous traits and binary traits under liability threshold model and case-control sampling

We have incorporated functions for a wide range of problems. Nevertheless, this largely remains as a preliminary work to be consolidated in the near future.
Test/Power calculation for mediating effect

Description

This function tests for or obtains power of mediating effect based on estimates of two regression coefficients and their standard errors. Note that for binary outcome or mediator, one should use log-odds ratio and its standard error.

Usage

ab(type, n=25000, a=0.15, sa=0.01, b=log(1.19), sb=0.01, alpha=0.05, fold=1)

Arguments

type string option: "test", "power"
n default sample size to be used for power calculation
a regression coefficient from independent variable to mediator
sa SE(a)
b regression coefficient from mediator variable to outcome
sb SE(b)
alpha size of significance test for power calculation
fold fold change for power calculation, as appropriate for a range of sample sizes

Value

The returned value are z-test and significance level for significant testing or sample size/power for a given fold change of the default sample size.
References

Author(s)
Jing Hua Zhao

See Also
ccsize

Examples

```r
## Not run:

ab()
n <- power <- vector()
for (j in 1:10)
{
  z <- ab(fold=j*0.01)
  n[j] <- z[1]
  power[j] <- z[2]
}
plot(n,power,xlab="Sample size",ylab="Power")
title("SNP-BMI-T2D association in EPIC-Norfolk study")

## End(Not run)
```

**Description**

This function is adapted from example 7.1 of Rabe-Hesketh et al. (2008). It also provides heritability estimate and confidence intervals.

**Usage**

```r
AE3(model, random, data, seed=1234, n.sim=50000, verbose=TRUE)
```
Arguments

model  a linear mixed model formula, see example below
random a random effect, see exampe below
data  data to be analyzed
seed  random number seed
n.sim  number of simulations
verbose a flag for printing out results

Value

The returned value is a list containing:

lme.result  the linear mixed model result
h2  the heritability estimate
CL  confidence intervals

References


Note

Adapted from f.mbf.R from the paper

Author(s)

Jing Hua Zhao

Examples

```r
## Not run:
AE3(bwt ~ male + first + midage + highage + birthyr,
    list(familyid = pdIdent(~var1 + var2 + var3 -1)), mfblong)
```

```r
## End(Not run)
```

### aldh2

**ALDH2 markers and Alcoholism**

Description

This data set contains eight ALDH2 markers and Japanese alcoholic patients (y=1) and controls (y=0). There are genotypes for 8 loci, with a prefix name (e.g., "EXON12") and a suffix for each of two alleles (".a1" and ".a2").
The eight markers loci follows the following map (base pairs)
D12S2070 (> 450 000),
D12S839 (> 450 000),
D12S821 (~ 400 000),
D12S1344 (~ 83 853),
EXON12 (~ 0),
EXON1 (~ 37 335),
D12S2263 (~ 38 927),
D12S1341 (> 450 000)

Usage
data(a1dh2)

Format
A data frame

Source
Prof Ian Craig of Oxford and SGDP Centre, KCL

References

Description
This data set contains APOE/APOC1 markers and Chinese Alzheimer’s patients and controls. Variable id is subject id and y takes value 0 for controls and 2 for Alzheimer’s.

The last six variables are age, sex and genotypes for APOE and APOC with suffixes for each of two alleles (.a1" and ".a2").

Usage
data(apoeapoc)

Format
A data frame
Source


---

**asplot**

*Regional association plot*

**Description**

This function obtains regional association plot for a particular locus, based on the information about recombination rates, linkage disequilibria between the SNP of interest and neighbouring ones, and single-point association tests p values.

Note that the best p value is not necessarily within locus in the original design.

**Usage**

```r
asplot(locus, map, genes, flanking=1e3, best_pval=NULL, sf=c(4,4), logpmax=10, pch=21)
```

**Arguments**

- **locus**: Data frame with columns c("CHR", "POS", "NAME", "PVAL", "RSQR") containing association results
- **map**: Genetic map, i.e, c("POS","THETA","DIST")
- **genes**: Gene annotation with columns c("START", "STOP", "STRAND", "GENE")
- **flanking**: Flanking length
- **best_pval**: Best p value for the locus of interest
- **sf**: scale factors for p values and recombination rates, smaller values are necessary for gene dense regions
- **logpmax**: Maximum value for -log10(p)
- **pch**: Plotting character for the SNPs to be highlighted, e.g., 21 and 23 refer to circle and diamond

**References**

DGI. Whole-genome association analysis identifies novel loci for type 2 diabetes and triglyceride levels. Science 2007;316(5829):1331-6

**Author(s)**

Paul de Bakker, Jing Hua Zhao, Shengxu Li
Examples

```r
# Not run:
asplot(CKN1locus, CDKMap, CDKNGenes)
title("CDK1A/CDK1B Region")
asplot(CKN1locus, CDKMap, CDKNGenes, best.pval=5.4e-8, sf=c(3,6))

# NCBI2R

options(stringsAsFactors=FALSE)
p <- with(CKN1locus, data.frame(SNP=NAME, PVAL))
hit <- subset(p, PVAL==min(PVAL, na.rm=TRUE))$SNP

library(NCBI2R)
# LD under build 36
pos <- apply(as.data.frame(p$SNP),1,GetSNPPosHapmap)
chr_pos <- do.call("rbind",pos)
l <- with(chr_pos, min(as.numeric(chrpos), na.rm=TRUE))
u <- with(chr_pos, max(as.numeric(chrpos), na.rm=TRUE))
LD <- with(chr_pos, GetLDInfo(unique(chr), l, u))
hit_LD <- subset(LD, SNP==hit)
hit_LD <- within(hit_LD, (RSQR=r2))
info <- GetSNPInfo(p$SNP)

haldane <- function(x) 0.5*(1-exp(-2*x))
locus <- with(info, data.frame(CHR=chr, POS=chrpos, NAME=marker,
                               DIST=(chrpos-min(chrpos))/1000000,
                               THETA=haldane((chrpos-min(chrpos))/100000000)))
locus <- merge.data.frame(locus, hit_LD, by="NAME", by.y="SNPB", all=TRUE)
locus <- merge.data.frame(locus, p, by="NAME", by.y="SNP", all=TRUE)
locus <- subset(locus, !is.na(POS))

ann <- AnnotateSNPList(p$SNP)
gen <- with(ann, data.frame(ID=locusID, CLASS=fxn_class, PATH=pathways,
                            START=GeneLowPoint, STOP=GeneHighPoint,
                            STRAND=ori, GENE=genesymbol, BUILD=build, CYTO=cyto))

attach(gen)

gen <- unique(GENE)
ustart <- as.vector(as.table(by(START, GENE, min))[ugen])
ustop <- as.vector(as.table(by(STOP, GENE, max))[ugen])
ustrand <- as.vector(as.table(by(as.character(STRAND), GENE, max))[ugen])
detach(gen)
gen <- data.frame(START=ustart, STOP=ustop, STRAND=ustrand, GENE=ugen)
gen <- subset(gen, START!=0)
rm(l,u,ugen,ustart,ustop,ustrand)
# Assume we have the latest map as in CDKMap
asplot(locus, CDKMap, gen)

# End(Not run)
```

---

Obtain correlation coefficients and their variance-covariances
**Description**

This function converts linear regression coefficients of phenotype on single nucleotide polymorphisms (SNPs) into Pearson correlation coefficients with their variance-covariance matrix. It is useful as a preliminary step for meta-analyze SNP-trait associations at a given region. Between-SNP correlations (e.g., from HapMap) are required as auxiliary information.

**Usage**

\[
b2r(b, s, rho, n)
\]

**Arguments**

- **b**: the vector of linear regression coefficients
- **s**: the corresponding vector of standard errors
- **rho**: triangular array of between-SNP correlation
- **n**: the sample size

**Value**

The returned value is a list containing:

- **r**: the vector of correlation coefficients
- **V**: the variance-covariance matrix of correlations

**References**


**Author(s)**

Jing Hua Zhao

**See Also**

mvmeta, LD22

**Examples**

```r
## Not run:
n <- 10
r <- c(1,0.2,1,0.4,0.5,1)
b <- c(0.1,0.2,0.3)
s <- c(0.4,0.3,0.2)
bs <- b2r(b,s,r,n)
```
BFDP

Bayesian false-discovery probability

Description
This function calculates BFDP, the approximate $P(H_0|\hat{\theta})$, given an estimate of the log relative risk, $\hat{\theta}$, the variance of this estimate, $V$, the prior variance, $W$, and the prior probability of a non-null association. When logscale=TRUE, the function accepts an estimate of the relative risk, $RR$, and the upper point of a 95% confidence interval $RR_{hi}$.

Usage
BFDP(a,b,pi,w,logscale=FALSE)

Arguments
- a: parameter value at which the power is to be evaluated
- b: the variance for a, or the upper point ($RR_{hi}$) of a 95%CI if logscale=FALSE
- pi: the prior probability of a non-null association
- w: the prior variance
- logscale: FALSE=the original scale, TRUE=the log scale

Value
The returned value is a list with the following components:
- PH0: probability given a,b
- PH1: probability given a,b,W
- BF: Bayes factor, $P_{H_0}/P_{H_1}$
- BFDP: Bayesian false-discovery probability
- ABF: approximate Bayes factor
- ABFDP: approximate Bayesian false-discovery probability

References

Note
adapted from BFDP functions by Jon Wakefield on 17th April, 2007
Author(s)
Jon Wakefield, Jing Hua Zhao

See Also
FPRP

Examples

```r
## Not run:

# Example from BDFP.xls by Jon Wakefield and Stephanie Monnier
# Step 1 - Pre-set an BFDP-level threshold for noteworthiness: BFDP values below this
# threshold are noteworthy
# The threshold is given by R/(1+R) where R is the ratio of the cost of a false
# non-discovery to the cost of a false discovery
T <- 0.8

# Step 2 - Enter up values for the prior that there is an association
pi0 <- c(0.7,0.5,0.01,0.001,0.00001,0.6)

# Step 3 - Enter the value of the OR that is the 97.5% point of the prior, for example
# if we pick the value 1.5 we believe that the prior probability that the
# odds ratio is bigger than 1.5 is 0.025.
ORhi <- 3

W <- (log(ORhi)/1.96)^2

# Step 4 - Enter OR estimate and 95% confidence interval (CI) to obtain BFDP

OR <- 1.316
OR_L <- 1.10
OR_U <- 2.50
logOR <- log(OR)
selogOR <- (log(OR_U)-log(OR))/1.96
r <- W/(W+selogOR^2)
z <- logOR/selogOR
z
ABF <- exp(-z^2*r/2)/sqrt(1-r)
ABF
FF <- (1-pi0)/pi0
FF
BFDPex <- FF*ABF/(FF*ABF+1)
BFDPex
pi0[BFDPex>T]

## now turn to BFDP
```
pi0 <- c(0.7, 0.5, 0.01, 0.0001, 0.6)
ORhi <- 3
OR <- 1.316
OR_U <- 2.56
W <- (log(ORhi)/1.96)^2
z <- BFDP(OR, OR_U, pi0, W)
z

bt

**Bradley-Terry model for contingency table**

**Description**

This function calculates statistics under Bradley-Terry model.

**Usage**

bt(x)

**Arguments**

x the data table

**Value**

The returned value is a list containing:

y A column of 1
count the frequency count/weight
allele the design matrix
bt.glm a glm.fit object
etdt.dat a data table that can be used by ETDT

**References**


**Note**

Adapted from a SAS macro for data in the example section

**Author(s)**

Jing Hua Zhao
Power and sample size for case-cohort design

### Description

The power of the test is according to

$$\Phi \left( Z_\alpha + m^{1/2} \theta \sqrt{ \frac{p_1 p_2 p_D}{q + (1-q)p_D} } \right)$$

where $\alpha$ is the significance level, $\theta$ is the log-hazard ratio for two groups, $p_j$, $j=1, 2$, are the proportion of the two groups in the population. $m$ is the total number of subjects in the subcohort, $p_D$ is the proportion of the failures in the full cohort, and $q$ is the sampling fraction of the subcohort.

Alternatively, the sample size required for the subcohort is

$$m = n B p_D / (n - B (1 - p_D))$$

where $B = (Z_{1-\alpha} + Z_\beta)^2 / (\theta^2 p_1 p_2 p_D)$, and $n$ is the size of cohort.

When infeasible configurations are specified, a sample size of -999 is returned.
Usage

csize(n,q,p0,p1,alpha,theta,power=NULL,verbose=FALSE)

Arguments

n the total number of subjects in the cohort
q the sampling fraction of the subcohort
p0 the proportion of the failures in the full cohort
p1 proportions of the two groups (p2=1-p1)
alpha significant level
theta log-hazard ratio for two groups
power if specified, the power for which sample size is calculated
verbose error messages are explicitly printed out

Value

The returned value is a value indicating the power or required sample size.

References


Note

Programmed for EPIC study

Author(s)

Jing Hua Zhao

See Also

pbsize

Examples

# Table 1 of Cai & Zeng (2004).
outfile <- "table1.txt"
cat("n","p0","p1","theta","q","power\n",file=outfile,sep="\t")
alpha <- 0.05
n <- 1000
for(p0 in c(0.10,0.05))
{
  for(p1 in c(0.3,0.5))
  {
    for(theta in c(0.5,1.0))
    {
      for(q in c(0.05,0.1))
      {
        for(power in c(0.8,0.9))
        {
          csize(n,q,p0,p1,alpha,theta,power)
for(q in c(0.1,0.2))
{
    power <- ccs(n,q,pD,p1,alpha,theta)
    cat(n,"\t",pD,"\t",p1,"\t",theta,"\t",q,"\t",signif(power,3),"\n",
         file=outfile,append=TRUE)
}

n <- 5000
for(pD in c(0.05,0.01))
{
    for(p1 in c(0.3,0.5))
    {
        for(theta in c(0.5,1.0))
        {
            for(q in c(0.01,0.02))
            {
                power <- ccs(n,q,pD,p1,alpha,theta)
                cat(n,"\t",pD,"\t",p1,"\t",theta,"\t",q,"\t",signif(power,3),"\n",
                     file=outfile,append=TRUE)
            }
        }
    }
}
for(i in 1:3)
{
    q <- s_nb[i]/n
    power <- ccs(n,q,pD,p1,alpha,log(theta[i]))
    ssize <- ccs(n,q,pD,p1,alpha,log(theta[i]),beta)
    cat(n,"\t",pD,"\t",p1,"\t",theta[i],"\t",q,"\t",signif(power,3),"\t",ssize,"\n",
        file=outfile,append=TRUE)
}
for(i in 1:3)
{
    q <- s_nb[i]/n
    power <- ccs(n,q,pD,p1,alpha,log(theta[i]))
    ssize <- ccs(n,q,pD,p1,alpha,log(theta[i]),beta)
    cat(n,"\t",pD,"\t",p1,"\t",theta[i],"\t",q,"\t",signif(power,3),"\t",ssize,"\n",
        file=outfile,append=TRUE)
}
for(i in 1:3)
{
    q <- s_nb[i]/n
    power <- ccs(n,q,pD,p1,alpha,log(theta[i]))
    ssize <- ccs(n,q,pD,p1,alpha,log(theta[i]),beta)
    cat(n,"\t",pD,"\t",p1,"\t",theta[i],"\t",q,"\t",signif(power,3),"\t",ssize,"\n",
        file=outfile,append=TRUE)
}

aric<-read.table(outfile,header=TRUE,sep="\t")
unlink(outfile)
# ARIC study
outfile <- "aric.txt"
for(Hq in cHPNQLPNRII)
{
    for(Hp in cHPNPULPNPQII)
    {
        for(Htheta in cHPNULQNPII)
        {
            for(Hq in cHPNPQLPNPRII)
            {
                power <- ccs(n,q,pD,p1,alpha,theta)
                cat(n,"\t",pD,"\t",p1,"\t",theta,"\t",q,"\t",signif(power,3),"\n",
                     file=outfile,append=TRUE)
            }
        }
    }
}

n <- 5000
for(pD in c(0.05,0.01))
{
    for(p1 in c(0.3,0.5))
    {
        for(theta in c(0.5,1.0))
        {
            for(q in c(0.01,0.02))
            {
                power <- ccs(n,q,pD,p1,alpha,theta)
                cat(n,"\t",pD,"\t",p1,"\t",theta,"\t",q,"\t",signif(power,3),"\n",
                     file=outfile,append=TRUE)
            }
        }
    }
}
for(i in 1:3)
{
    q <- s_nb[i]/n
    power <- ccs(n,q,pD,p1,alpha,log(theta[i]))
    ssize <- ccs(n,q,pD,p1,alpha,log(theta[i]),beta)
    cat(n,"\t",pD,"\t",p1,"\t",theta[i],"\t",q,"\t",signif(power,3),"\t",ssize,"\n",
        file=outfile,append=TRUE)
}

aric<-read.table(outfile,header=TRUE,sep="\t")
unlink(outfile)
# EPIC study
outfile <- "epic.txt"
for(Hq in cHPNQLPNRII)
{
    for(Htheta in cHPNULQNPII)
    {
        for(Hq in cHPNPQLPNPRII)
        {
            power <- ccs(n,q,pD,p1,alpha,theta)
            cat(n,"\t",pD,"\t",p1,"\t",theta,"\t",q,"\t",signif(power,3),"\n",
                 file=outfile,append=TRUE)
        }
    }

n <- 5000
for(pD in c(0.3,0.2,0.1,0.05))
{
    for(p1 in c(0.3,0.5))
    {
        for(theta in c(0.5,1.0))
        {
            for(q in c(0.01,0.02))
            {
                power <- ccs(n,q,pD,p1,alpha,theta)
                cat(n,"\t",pD,"\t",p1,"\t",theta,"\t",q,"\t",signif(power,3),"\n",
                     file=outfile,append=TRUE)
            }
        }
    }
}

for(i in 1:3)
{
    q <- s_nb[i]/n
    power <- ccs(n,q,pD,p1,alpha,log(theta[i]))
    ssize <- ccs(n,q,pD,p1,alpha,log(theta[i]),beta)
    cat(n,"\t",pD,"\t",p1,"\t",theta[i],"\t",q,"\t",signif(power,3),"\t",ssize,"\n",
        file=outfile,append=TRUE)
}

aric<-read.table(outfile,header=TRUE,sep="\t")
unlink(outfile)
# EPIC study
outfile <- "epic.txt"
Cystic fibrosis data

Description

This data set contains a case-control indicator and 23 SNPs.

The inter-marker distances (Morgan) are as follows

0.000090, 0.000158, 0.005000, 0.000100, 0.000200, 0.000150, 0.000250, 0.000200, 0.000050, 0.000350, 0.000300, 0.000250, 0.000350, 0.000350, 0.000800, 0.000100, 0.000200, 0.000150, 0.000550, 0.006000, 0.000700, 0.001000
Usage

data(cf)

Format

A data frame containing 186 rows and 24 columns

Note

This can be used as an example of converting PL-EM to matrix format,

cfdata <- vector("numeric")
cfname <- vector("character")
for (i in 2:dim(cf)[2])
{
  tmp <- plem2m(cf[,i])
a1 <- tmp[[1]]
a2 <- tmp[[2]]
cfdata <- cbind(cfdata,a1,a2)
a1name <- paste("loc",i-1,".a1",sep="")
a2name <- paste("loc",i-1,".a2",sep="")
cfname <- cbind(cfname,a1name,a2name)
}
cfdata <- as.data.frame(cfdata)
names(cfdata) <- cfname

Source


Description

Chow's test is for differences between two or more regressions. Assuming that errors in regressions 1 and 2 are normally distributed with zero mean and homoscedastic variance, and they are independent of each other, the test of regressions from sample sizes \( n_1 \) and \( n_2 \) is then carried out using the following steps. 1. Run a regression on the combined sample with size \( n = n_1 + n_2 \) and obtain within group sum of squares called \( S_1 \). The number of degrees of freedom is \( n_1 + n_2 - k \), with \( k \) being the number of parameters estimated, including the intercept. 2. Run two regressions on the two individual samples with sizes \( n_1 \) and \( n_2 \), and obtain their within group sums of square \( S_2 + S_3 \), with \( n_1 + n_2 - 2k \) degrees of freedom. 3. Conduct an \( F(k,n_1+n_2-2k) \) test defined by

\[
F = \frac{[S_1 - (S_2 + S_3)]/k}{[(S_2 + S_3)/(n_1 + n_2 - 2k)]}
\]
If the $F$ statistic exceeds the critical $F$, we reject the null hypothesis that the two regressions are equal. In the case of haplotype trend regression, haplotype frequencies from combined data are known, so can be directly used.

**Usage**

```r
chow.test(y1,x1,y2,x2,x=NULL)
```

**Arguments**

- `y1` a vector of dependent variable
- `x1` a matrix of independent variables
- `y2` a vector of dependent variable
- `x2` a matrix of independent variables
- `x` a known matrix of independent variables

**Value**

The returned value is a vector containing (please use subscript to access them):

- `F` the F statistic
- `df1` the numerator degree(s) of freedom
- `df2` the denominator degree(s) of freedom
- `p` the p value for the F test

**References**


**Note**

adapted from chow.R

**Author(s)**

Shigenobu Aoki, Jing Hua Zhao

**Source**

[http://aoki2.si.gunma-u.ac.jp/R/](http://aoki2.si.gunma-u.ac.jp/R/)

**See Also**

- `htr`
Examples

```r
## Not run:
dat1 <- matrix(c(  
1.2, 1.9, 0.9,  
1.6, 2.7, 1.3,  
3.5, 3.7, 2.0,  
4.0, 3.1, 1.8,  
5.6, 3.5, 2.2,  
5.7, 7.5, 3.5,  
6.7, 1.2, 1.9,  
7.5, 3.7, 2.7,  
8.5, 0.6, 2.1,  
9.7, 5.1, 3.6), byrow=TRUE, ncol=3)

dat2 <- matrix(c(  
1.4, 1.3, 0.5,  
1.5, 2.3, 1.3,  
3.1, 3.2, 2.5,  
4.4, 3.6, 1.1,  
5.1, 3.1, 2.8,  
5.2, 7.3, 3.3,  
6.5, 1.5, 1.3,  
7.8, 3.2, 2.2,  
8.1, 0.1, 2.8,  
9.5, 5.6, 3.9), byrow=TRUE, ncol=3)

y1<-dat1[,3]  
y2<dat2[,3]  
x1<-dat1[,1:2]  
x2<-dat2[,1:2]  
chow.test.r<-chow.test(y1,x1,y2,x2)

## End(Not run)
```

### Description

The function empirically estimate the variance of the score functions. The variance-covariance matrix consists of two parts: the additive part and the part for the individual-specific environmental effect. Other reasonable decompositions are possible.

This program has the following improvement over "score.r":

1. It works with selected nuclear families
2. Trait data on parents (one parent or two parents), if available, are utilized.
3. Besides a statistic assuming no locus-specific dominance effect, it also computes a statistic that allows for such effect. It computes two statistics instead of one.

Function "merge" is used to merge the IBD data for a pair with the transformed trait data (i.e., \( w_kw_l \)).
Usage

comp.score(ibddata="ibd_dist.out", phenotype="pheno.dat", mean=0,
var=1, h2=0.3)

Arguments

ibddata The output file from GENEHUNTER using command "dump ibd". The default file name is ibd_dist.out.

phenotype The file of pedigree structure and trait value. The default file name is "pheno.dat". Columns (no headings) are: family ID, person ID, father ID, mother ID, gender, trait value, where Family ID and person ID must be numbers, not characters. Use character "NA" for missing phenotypes.

mean (population) mean of the trait, with a default value of 0

var (population) variance of the trait, with a default value of 1

h2 heritability of the trait, with a default value of 0.3

Value

a matrix with each row containing the location and the statistics and their p-values.

References


Note

Adapt from score2.r

Author(s)

Yingwei Peng, Kai Wang

Examples

```R
# Not run:
# An example based on GENEHUNTER version 2.1, with quantitative trait data in file
# "pheno.dat" generated from the standard normal distribution. The following
# example shows that it is possible to automatically call GENEHUNTER using R
# function "system".

cwd <- getwd()
cs.dir <- file.path(path.package("gap"),"tests/comp.score")
setwd(cs.dir)
```
Data Description

The data set consist of 103 common (>5% minor allele frequency) SNPs genotyped in 129 trios from an European-derived population. These SNPs are in a 500-kb region on human chromosome 5q31 implicated as containing a genetic risk factor for Crohn disease.

The positions, names and haplotype blocks reported are as follows,

274044 IGR1118a_1 BLOCK 1
274541 IGR1119a_1 *
286593 IGR1143a_1 *
287261 IGR1144a_1 *
299755 IGR1169a_2 *
324341 IGR1218a_2 *
324379 IGR1219a_2 *
358048 IGR1286a_1 BLOCK 1
366811 TSC0101718
395079 IGR1373a_1 BLOCK 2
396353 IGR1371a_1 *
397334 IGR1369a_2 *
397381 IGR1369a_1 *
398352 IGR1367a_1 BLOCK 2
411823 IGR2008a_2
411873 IGR2008a_1 BLOCK 3
412456 IGR2010a_3 *
413233 IGR2011b_1 *
415579 IGR2016a_1 *
417617 IGR2020a_15 *
419845 IGR2025a_2 *
424283 IGR2033a_1 *
425376 IGR2036a_2 *
425549 IGR2036a_1 BLOCK 3
433467 IGR2052a_1 BLOCK 4
435282 IGR2055a_1 *
437682 IGR2060a_1 *
438883 IGR2063b_1 *
443565 IGR2072a_2 *
443750 IGR2073a_1 *
<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>445337</td>
<td>IGR2076a_1</td>
<td></td>
</tr>
<tr>
<td>447791</td>
<td>IGR2081a_1</td>
<td></td>
</tr>
<tr>
<td>449895</td>
<td>IGR2085a_2</td>
<td></td>
</tr>
<tr>
<td>455246</td>
<td>IGR2096a_1</td>
<td></td>
</tr>
<tr>
<td>463136</td>
<td>IGR2111a_3 BLOCK 4</td>
<td></td>
</tr>
<tr>
<td>482171</td>
<td>IGR2150a_1 BLOCK 5</td>
<td></td>
</tr>
<tr>
<td>485828</td>
<td>IGR2157a_1</td>
<td></td>
</tr>
<tr>
<td>495082</td>
<td>IGR2175a_2</td>
<td></td>
</tr>
<tr>
<td>506266</td>
<td>IGR2198a_1</td>
<td></td>
</tr>
<tr>
<td>506890</td>
<td>IGR2199a_1 BLOCK 5</td>
<td></td>
</tr>
<tr>
<td>507208</td>
<td>IGR2200a_1 BLOCK 6</td>
<td></td>
</tr>
<tr>
<td>508338</td>
<td>IGR2202a_1</td>
<td></td>
</tr>
<tr>
<td>508858</td>
<td>IGR2203a_1</td>
<td></td>
</tr>
<tr>
<td>510951</td>
<td>IGR2207a_1</td>
<td></td>
</tr>
<tr>
<td>518478</td>
<td>IGR2222a_2 BLOCK 6</td>
<td></td>
</tr>
<tr>
<td>519387</td>
<td>IGR2224a_2 BLOCK 7</td>
<td></td>
</tr>
<tr>
<td>519962</td>
<td>IGR2225a_1</td>
<td></td>
</tr>
<tr>
<td>520521</td>
<td>IGR2226a_3</td>
<td></td>
</tr>
<tr>
<td>522600</td>
<td>IGR2230a_1</td>
<td></td>
</tr>
<tr>
<td>525243</td>
<td>IGR2236a_1</td>
<td></td>
</tr>
<tr>
<td>529556</td>
<td>IGR2244a_4</td>
<td></td>
</tr>
<tr>
<td>532363</td>
<td>IGR2250a_4</td>
<td></td>
</tr>
<tr>
<td>545062</td>
<td>IGR2276a_1</td>
<td></td>
</tr>
<tr>
<td>553189</td>
<td>IGR2292a_1</td>
<td></td>
</tr>
<tr>
<td>570978</td>
<td>IGR3005a_1</td>
<td></td>
</tr>
<tr>
<td>571022</td>
<td>IGR3005a_2</td>
<td></td>
</tr>
<tr>
<td>576586</td>
<td>IGR3016a_1</td>
<td></td>
</tr>
<tr>
<td>577141</td>
<td>IGR3018a_2</td>
<td></td>
</tr>
<tr>
<td>577838</td>
<td>IGR3019a_2</td>
<td></td>
</tr>
<tr>
<td>578122</td>
<td>IGR3020a_1</td>
<td></td>
</tr>
<tr>
<td>579217</td>
<td>IGR3022a_1</td>
<td></td>
</tr>
<tr>
<td>579529</td>
<td>IGR3023a_1</td>
<td></td>
</tr>
<tr>
<td>579818</td>
<td>IGR3023a_3</td>
<td></td>
</tr>
<tr>
<td>582651</td>
<td>IGR3029a_1</td>
<td></td>
</tr>
<tr>
<td>582948</td>
<td>IGR3029a_2</td>
<td></td>
</tr>
<tr>
<td>583131</td>
<td>IGR3030a_1</td>
<td></td>
</tr>
<tr>
<td>587836</td>
<td>IGR3039a_1</td>
<td></td>
</tr>
<tr>
<td>590425</td>
<td>IGR3044a_1</td>
<td></td>
</tr>
<tr>
<td>590585</td>
<td>IGR3045a_1</td>
<td></td>
</tr>
<tr>
<td>594115</td>
<td>IGR3051a_1</td>
<td></td>
</tr>
<tr>
<td>594812</td>
<td>IGR3053a_1</td>
<td></td>
</tr>
<tr>
<td>598805</td>
<td>IGR3061a_1</td>
<td></td>
</tr>
<tr>
<td>601294</td>
<td>IGR3066a_1</td>
<td></td>
</tr>
<tr>
<td>608759</td>
<td>IGR3081a_1</td>
<td></td>
</tr>
<tr>
<td>610447</td>
<td>IGR3084a_1</td>
<td></td>
</tr>
<tr>
<td>611177</td>
<td>IGR3086a_1 BLOCK 7</td>
<td></td>
</tr>
<tr>
<td>613488</td>
<td>IGR3090a_1</td>
<td></td>
</tr>
<tr>
<td>616241</td>
<td>IGR3096a_1 BLOCK 8</td>
<td></td>
</tr>
</tbody>
</table>
However it has been updated after the paper was published (posted on http://www.broad.mit.edu/humgen/IBD5/haploData.html)


Usage

data(crohn)

Format

A data frame containing 387 rows and 212 columns

Source

Description

The function accepts parameter estimates and their standard errors for a range of models.

Usage

\texttt{ESplot(ESdat, SE=TRUE, logscale=TRUE, alpha=0.05, xlim=c(-2,8), v=1,...)}

Arguments

- **ESdat**: A data frame consisting of model id, parameter estimates and standard errors or confidence limits
- **SE**: If TRUE, the third column of ESdata contains the standard error estimates
- **logscale**: If TRUE, indicates log-scale as appropriate for odds ratio
- **alpha**: Type-I error rate used to construct 100(1-alpha) confidence interval
- **xlim**: Lower and upper limits of the horizontal axis, roughly corresponding to confidence limits
- **...**: Other options for \texttt{plot}
- **v**: Location of the vertical line

Author(s)

Jing Hua Zhao

Examples

```R
## Not run:
# 7-4-2008 MRC-Epid JHZ
options(stringsAsFactors=FALSE)
testdata <- data.frame(models=c("Basic model","Adjusted","Moderately adjusted",
                              "Heavily adjusted","Other"),
                      OR = c(4.5,3.5,2.5,1.5,1),
                      SElOGOR = c(0.2,0.1,0.5,0.5,0.2))
ESplot(testdata,v=1)
title("This is a fictitious plot")
```

```R
# Quantitative trait, as appropriate for linear regression
# testdata <- data.frame(modelid, beta, se(beta))
# ESplot(testdata, logscale=FALSE)

# Other scenarios
# OR with CI
# ESplot(testdata,SE=FALSE)
```

## End(Not run)
**Friedreich Ataxia data**

**Description**

This data set contains a case-control indicator and twelve microsatellite markers. An extra unphased individual with the following genotype

\[
\begin{array}{cccccccccccc}
2 & 7 & 7 & 7 & 1 & 3 & 2 & 2 & 2 & 6 & 3 \\
3 & 8 & 10 & 8 & 3 & 9 & 3 & 4 & 2 & 2 & 7 & 5
\end{array}
\]

has not been included.

The inter-marker distances (Morgan) are as follows,

0.03, 0.065, 0.00125, 0.00125, 0.00125, 0.00125, 0.00125, 0.00125, 0.0125, 0.0125, 0.0125, 0.0125, 0.045

**Usage**

data(fa)

**Format**

A data frame containing 127 rows and 13 columns

**Source**


---

**Sample size for family-based linkage and association design**

**Description**

This function implements Risch and Merikangas (1996) statistics evaluating power for family-based linkage (affected sib pairs, ASP) and association design. They are potentially useful in the prospect of genome-wide association studies.

The function calls auxiliary functions sn() and strlen; sn() contains the necessary thresholds for power calculation while strlen() evaluates length of a string (generic).

**Usage**

fbsize(gamma,p,alpha=c(1e-4,1e-8,1e-8),beta=0.2,debug=0,error=0)
Arguments

- **gamma**: genotype relative risk assuming multiplicative model
- **p**: frequency of disease allele
- **alpha**: Type I error rates for ASP linkage, TDT and ASP-TDT
- **beta**: Type II error rate
- **debug**: verbose output
- **error**: 0=use the correct formula, 1=the original paper

Value

The returned value is a list containing:

- **gamma**: input gamma
- **p**: input p
- **n1**: sample size for ASP
- **n2**: sample size for TDT
- **n3**: sample size for ASP-TDT
- **lambda0**: lambda o
- **lambdaS**: lambda s

References


Note

extracted from rm.c

Author(s)

Jing Hua Zhao

See Also

pbsize
### Examples

```r
models <- matrix(c(
  4.0, 0.01,
  4.0, 0.10,
  4.0, 0.50,
  4.0, 0.80,
  2.0, 0.01,
  2.0, 0.10,
  2.0, 0.50,
  2.0, 0.80,
  1.5, 0.01,
  1.5, 0.10,
  1.5, 0.50,
  1.5, 0.80), ncol=2, byrow=TRUE)
outfile <- "fbsize.txt"
cat("gamma", "p", "n", "nasp", "P_A", "H1", "N_tdt", "H2", "N_asp/tdt", "L_o", "L_s\n", file=outfile, sep="\t")
for(i in 1:12) {
  g <- models[i,1]
  p <- models[i,2]
  z <- fbsize(g,p)
  cat("\n\n\ngamma, z=p, z=y, z=n1, z=pA, z=h1, z=n2, z=h2, z=n3, z=\lambda_o, z=\lambda_s, file=\nfile=\nappend=\ntrue, sep="\t")
  cat("\n\n\n\n\n"file=outfile, append=TRUE)
}
table1 <- read.table(outfile, header=TRUE, sep="\t")
nc <- c(4,7,9)
table1[,nc] <- ceiling(table1[,nc])
dc <- c(3,5,6,8,10,11)
table1[,dc] <- round(table1[,dc],2)
unlink(outfile)
# APOE-4, Scott WK, Pericak-Vance, MA & Haines JL
# Genetic analysis of complex diseases 1327
# g <- 4.5
# p <- 0.15
cat("\n\n\n\n\n\nAlzheimer's:\n\n")
fbsize(g,p)
# note to replicate the Table we need set alpha=9.961139e-05,4.910638e-08 and
# beta=0.2004542 or reset the quantiles in fbsize.R
```

### Description

The function calculates the false positive report probability (FPRP), the probability of no true association between a genetic variant and disease given a statistically significant finding, which depends not only on the observed P value but also on both the prior probability that the association is real and the statistical power of the test. An associate result is the false negative reported probability (FNRP). See example for the recommended steps.
The FPRP and FNRP are derived as follows. Let \( H_0 = \) null hypothesis (no association), \( H_A = \) alternative hypothesis (association). Since classic frequentist theory considers they are fixed, one has to resort to Bayesian framework by introducing prior, \( \pi = P(H_0 = TRUE) = P(\text{association}). \) Let \( T = \) test statistic, and \( P(T > z_\alpha | H_0 = TRUE) = P(\text{rejecting } H_0 | H_0 = TRUE) = \alpha, \) \( P(T > z_\alpha | H_0 = FALSE) = P(\text{rejecting } H_0 | H_A = TRUE) = 1 - \beta. \) The joint probability of test and truth of hypothesis can be expressed by \( \alpha, \beta \) and \( \pi. \)

<table>
<thead>
<tr>
<th>Truth of ( H_A )</th>
<th>significant</th>
<th>nonsignificant</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUE</td>
<td>( (1 - \beta)\pi )</td>
<td>( \beta\pi )</td>
<td>( \pi )</td>
</tr>
<tr>
<td>FALSE</td>
<td>( \alpha(1 - \pi) )</td>
<td>( (1 - \alpha)(1 - \pi) )</td>
<td>( 1 - \pi )</td>
</tr>
<tr>
<td>Total</td>
<td>( (1 - \beta)\pi + \alpha(1 - \pi) )</td>
<td>( \beta\pi + (1 - \alpha)(1 - \pi) )</td>
<td>( 1 )</td>
</tr>
</tbody>
</table>

We have \( FPRP = P(H_0 = TRUE | T > z_\alpha) = \alpha(1 - \pi)/[\alpha(1 - \pi) + (1 - \beta)\pi] = \{1 + \pi/(1 - \pi)\}[(1 - \beta)/\alpha]^{-1} \) and similarly \( FNRP = \{1 + [(1 - \alpha)/\beta][(1 - \pi)/\pi] \}^{-1}. \)

**Usage**

\[
FPRP(a,b,pi0,ORlist,logscale=FALSE)
\]

**Arguments**

- \( a \) parameter value at which the power is to be evaluated
- \( b \) the variance for \( a \), or the upper point of a 95% CI if \( \text{logscale=FALSE} \)
- \( \text{pi0} \) the prior probability that \( H_0 \) is true
- \( \text{ORlist} \) a vector of ORs that is most likely
- \( \text{logscale} \) FALSE\( =a,b \) in original scale, TRUE\( =a,b \) in log scale

**Value**

The returned value is a list with components,

- \( p \) \( = \) p value corresponding to \( a,b \)
- \( \text{power} \) the power corresponding to the vector of ORs
- \( \text{FPRP} \) False-positive report probability
- \( \text{FNRP} \) False-negative report probability

**References**


**Author(s)**

Jing Hua Zhao
See Also

BFDP

Examples

```r
## Not run:
# Example by Laure El ghormli & Sholom Wacholder on 25-Feb-2004
# Step 1 - Pre-set an FPRP-level criterion for noteworthiness

T <- 0.2

# Step 2 - Enter values for the prior that there is an association
pi0 <- c(0.25, 0.1, 0.01, 0.001, 0.0001, 0.00001)

# Step 3 - Enter values of odds ratios (OR) that are most likely, assuming that
# there is a non-null association
ORlist <- c(1.2, 1.5, 2.0)

# Step 4 - Enter OR estimate and 95

OR <- 1.316
ORlo <- 1.08
ORhi <- 1.60

logOR <- log(OR)
selogOR <- abs(logOR - log(ORhi))/1.96

p <- ifelse(logOR > 0, 2 * (1 - pnorm(logOR / selogOR)), 2 * pnorm(logOR / selogOR))

q <- qnorm(1 - p/2)
POWER <- ifelse(log(ORlist) > 0, 1 - pnorm(q - log(ORlist) / selogOR),
               pnorm(-q - log(ORlist) / selogOR))

POWER

FPRPex <- t(p*(1-pi0)/(p*(1-pi0)+POWER%o%pi0))
rownames(FPRPex) <- pi0
colnames(FPRPex) <- ORlist
FPRPex
FPRPex > T

## now turn to FPRP

OR <- 1.316
ORhi <- 1.60
ORlist <- c(1.2, 1.5, 2.0)
pi0 <- c(0.25, 0.1, 0.01, 0.001, 0.0001, 0.00001)
z <- FPRP(OR, ORhi, pi0, ORlist, logscale=FALSE)
z

## End(Not run)
```
fsnps

A case-control data involving four SNPs with missing genotype

Description
This is a simulated data of four SNPs with their alleles coded in characters. The variable y contains phenotypes (1=case, 0=control).

Usage
data(fsnps)

Format
A data frame

Source
Dr Sebastien Lissarrague of Genset

gc.em
Gene counting for haplotype analysis

Description
Gene counting for haplotype analysis with missing data, adapted for hap.score

Usage
gc.em(data, locus.label=NA, converge.eps=1e-06, maxiter=500, handle.miss=0, miss.val=0, control=gc.control())

Arguments
data Matrix of alleles, such that each locus has a pair of adjacent columns of alleles, and the order of columns corresponds to the order of loci on a chromosome. If there are K loci, then ncol(data) = 2*K. Rows represent alleles for each subject.
locus.label Vector of labels for loci, of length K (see definition of data matrix).
converge.eps Convergence criterion, based on absolute change in log likelihood (lnlike).
maxiter Maximum number of iterations of EM.
handle.miss a flag for handling missing genotype data, 0=no, 1=yes
miss.val missing value
control a function, see genecounting
Value

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>converge</td>
<td>Indicator of convergence of the EM algorithm (1=converged, 0 = failed).</td>
</tr>
<tr>
<td>niter</td>
<td>Number of iterations completed in the EM algorithm.</td>
</tr>
<tr>
<td>locus.info</td>
<td>A list with a component for each locus. Each component is also a list, and</td>
</tr>
<tr>
<td></td>
<td>the items of a locus-specific list are the locus name and a vector for the</td>
</tr>
<tr>
<td></td>
<td>unique alleles for the locus.</td>
</tr>
<tr>
<td>locus.label</td>
<td>Vector of labels for loci, of length K (see definition of input values).</td>
</tr>
<tr>
<td>haplotype</td>
<td>Matrix of unique haplotypes. Each row represents a unique haplotype, and</td>
</tr>
<tr>
<td></td>
<td>the number of columns is the number of loci.</td>
</tr>
<tr>
<td>hap.prob</td>
<td>Vector of mle’s of haplotype probabilities. The ith element of hap.prob</td>
</tr>
<tr>
<td></td>
<td>corresponds to the ith row of haplotype.</td>
</tr>
<tr>
<td>hap.prob.nold</td>
<td>Similar to hap.prob, but assuming no linkage disequilibrium.</td>
</tr>
<tr>
<td>lnlike</td>
<td>Value of lnlike at last EM iteration (maximum lnlike if converged).</td>
</tr>
<tr>
<td>lr</td>
<td>Likelihood ratio statistic to test no linkage disequilibrium among all loci.</td>
</tr>
<tr>
<td>indx.subj</td>
<td>Vector for index of subjects, after expanding to all possible pairs of</td>
</tr>
<tr>
<td></td>
<td>haplotypes for each person. If indx=i, then i is the ith row of input</td>
</tr>
<tr>
<td></td>
<td>matrix data. If the ith subject has n possible pairs of haplotypes that</td>
</tr>
<tr>
<td></td>
<td>correspond to their marker phenotype, then i is repeated n times.</td>
</tr>
<tr>
<td>nreps</td>
<td>Vector for the count of haplotype pairs that map to each subject’s marker</td>
</tr>
<tr>
<td></td>
<td>genotypes.</td>
</tr>
<tr>
<td>hap1code</td>
<td>Vector of codes for each subject’s first haplotype. The values in hap1code</td>
</tr>
<tr>
<td></td>
<td>are the row numbers of the unique haplotypes in the returned matrix</td>
</tr>
<tr>
<td></td>
<td>haplotype.</td>
</tr>
<tr>
<td>hap2code</td>
<td>Similar to hap1code, but for each subject’s second haplotype.</td>
</tr>
<tr>
<td>post</td>
<td>Vector of posterior probabilities of pairs of haplotypes for a person,</td>
</tr>
<tr>
<td></td>
<td>given their marker phenotypes.</td>
</tr>
<tr>
<td>htrtable</td>
<td>A table which can be used in haplotype trend regression</td>
</tr>
</tbody>
</table>

References


Note

Adapted from GENECOUNTING

Author(s)

Jing Hua Zhao
See Also

genecounting, LDkl

Examples

## Not run:
data(hla)
gc.em(hla[,3:8],locus.label=c("DQR","DQA","DQB"),control=gc.control(assignment="t"))

## End(Not run)

gcontrol  genometric control

Description

The Bayesian genomic control statistics with the following parameters,

- **n**: number of loci under consideration
- **lambdahat**: median of the n trend statistics/0.46
- **Prior** for noncentrality parameter Ai is Normal(sqrt(lambdahat)kappa,lambdahat*tau2)
- **kappa**: multiplier in prior above, set at 1.6 * sqrt(log(n))
- **tau2**: multiplier in prior above
- **epsilon**: prior probability a marker is associated, set at 10/n
- **ngib**: number of cycles for the Gibbs sampler after burn in
- **burn**: number of cycles for the Gibbs sampler to burn in

Armitage’s trend test along with the posterior probability that each marker is associated with the disorder is given. The latter is not a p-value but any value greater than 0.5 (pout) suggests association.

Usage

gcontrol(data,zeta,kappa,tau2,epsilon,ngib,burn,idum)

Arguments

data  the data matrix
zeta  program constant with default value 1000
kappa  multiplier in prior for mean with default value 4
tau2  multiplier in prior for variance with default value 1
epsilon  prior probability of marker association with default value 0.01
ngib  number of Gibbs steps, with default value 500
burn  number of burn-ins with default value 50
idum  seed for pseudorandom number sequence
Value

The returned value is a list containing:

deltot  the probability of being an outlier
x2    the $\chi^2$ statistic
A     the A vector

References


Note

Adapted from gcontrol by Bobby Jones and Kathryn Roeder, use -Dexecutable for standalone program, function getnum in the original code needs %*s to skip id string

Author(s)

Bobby Jones, Jing Hua Zhao

Source

http://www.stat.cmu.edu

Examples

```r
## Not run:
test<-c(1,2,3,4,5,6, 1,2,1,23,1,2, 100,1,2,12,1,1,
1,2,3,4,5,6,1,2,11,23,1,2, 10,11,2,12,1,11)
test<-matrix(test,nrow=6,byrow=T)
gcontrol(test)

## End(Not run)
```

Description

The function obtains 1-df $\chi^2$ statistics (observed) according to a vector of p values, and the inflation factor (lambda) according to medians of the observed and expected statistics. The latter is based on the empirical distribution function (EDF) of 1-df $\chi^2$ statistics.

It would be appropriate for genetic association analysis as of 1-df Armitage trend test for case-control data; for 1-df additive model with continuous outcome one has to consider the compatibility with p values based on z/t statistics.
### gcp

**Permutation tests using GENECOUNTING**

**Description**

This function is a R port of the GENECOUNTING/PERMUTE program which generates EHPLUS-type statistics including z-tests for individual haplotypes.

**Usage**

```r
 gcp(y, cc, g, handle.miss=1, miss.val=0, n.sim=0,
    locus.label=NULL, quietly=FALSE)
```

---

**Usage**

```r
 gcontrol2(p, col=palette()[4], lcol=palette()[2],...)
```

**Arguments**

- `p`: a vector of observed p values
- `col`: colour for points in the Q-Q plot
- `lcol`: colour for the diagonal line in the Q-Q plot
- `...`: other options for plot

**Value**

A list containing:

- `x`: the expected $\chi^2$ statistics
- `y`: the observed $\chi^2$ statistics
- `lambda`: the inflation factor

**References**


**Author(s)**

Jing Hua Zhao

**Examples**

```r
 ## Not run:
 x2 <- rchisq(100,1,1)
 p <- pchisq(x2,1,lower.tail=FALSE)
 r <- gcontrol2(p)
 print(r$lambda)

 ## End(Not run)
```
Arguments

- `y`: A column of 0/1 indicating cases and controls
- `cc`: Analysis indicator, 0 = marker-marker, 1 = case-control
- `g`: The multilocus genotype data
- `handle.miss`: A flag with value 1 indicating missing data are allowed
- `miss.val`: Missing value
- `n.sim`: The number of permutations
- `locus.label`: Label of each locus
- `quietly`: A flag if TRUE will suppress the screen output

Value

The returned value is a list containing (p.sim and ph when n.sim > 0):

- `x2obs`: The observed chi-squared statistic
- `pobs`: The associated p value
- `zobs`: The observed z value for individual haplotypes
- `p.sim`: Simulated p value for the global chi-squared statistic
- `ph`: Simulated p values for individual haplotypes

References


Note

Built on gcp.c

Author(s)

Jing Hua Zhao

See Also

genecounting
Examples

```r
## Not run:
data(fsnps)
y<-fsnps$y
c<-1
g<-fsnps[,3:10]

gcp(y,cc,g,miss.val="Z",n.sim=5)
hap.score(y,g,method="hap",miss.val="Z")
```

## End(Not run)

---

**Description**

Gene counting for haplotype analysis with missing data

**Usage**

```r
genecounting(data,weight=NULL,loci=NULL,control=gc.control())
```

**Arguments**

- **data**: genotype table
- **weight**: a column of frequency weights
- **loci**: an array containing number of alleles at each locus
- **control**: is a function with the following arguments:
  1. **xdata**: a flag indicating if the data involves X chromosome, if so, the first column of data indicates sex of each subject: 1=male, 2=female. The marker data are no different from the autosomal version for females, but for males, two copies of the single allele present at a given locus.
  2. **convll**: set convergence criteria according to log-likelihood, if its value set to 1
  3. **handle.miss**: to handle missing data, if its value set to 1
  4. **eps**: the actual convergence criteria, with default value 1e-5
  5. **tol**: tolerance for genotype probabilities with default value 1e-8
  6. **maxit**: maximum number of iterations, with default value 50
  7. **pl**: criteria for trimming haplotypes according to posterior probabilities
  8. **assignment**: filename containing haplotype assignment
  9. **verbose**: If TRUE, yields print out from the C routine
Value

The returned value is a list containing:

- \( h \) haplotype frequency estimates under linkage disequilibrium (LD)
- \( h0 \) haplotype frequency estimates under linkage equilibrium (no LD)
- \( prob \) genotype probability estimates
- \( l0 \) log-likelihood under linkage equilibrium
- \( l1 \) log-likelihood under linkage disequilibrium
- \( hapid \) unique haplotype identifier (defunct, see gc.em)
- \( npusr \) number of parameters according user-given alleles
- \( npdat \) number of parameters according to observed
- \( htrtable \) design matrix for haplotype trend regression (defunct, see gc.em)
- \( iter \) number of iterations used in gene counting
- \( converge \) a flag indicating convergence status of gene counting
- \( di0 \) haplotype diversity under no LD, defined as \( 1 - \sum(h_0^2) \)
- \( di1 \) haplotype diversity under LD, defined as \( 1 - \sum(h^2) \)
- \( resid \) residuals in terms of frequency weights = o - e

References


Note

adapted from GENECOUNTING

Author(s)

Jing Hua Zhao

See Also

gc.em, LDkl
Examples

```r
## Not run:
## HLA data
data(hla)
充实(hla[,3:8])
summary(hla.gc)
充实gc$l0
充实gc$l1

## ALDH2 data
data(aldh2)
control <- gc.control(handle.miss=1,assignment="ALDH2.out")
aldh2.gc <- genecounting(aldh2[,3:6],control=control)
summary(aldh2.gc)
aldh2.gc$l0
aldh2.gc$l1

## Chromosome X data
# assuming allelic data have been extracted in columns 3-13
# and column 3 is sex
filespec <- system.file("tests/genecounting/mao.dat")
mao2 <- read.table(filespec)
data <- mao2[,3:13]
loci <- c(12,9,6,5,3)
contr <- gc.control(xdata=TRUE,handle.miss=1)
mao.gc <- genecounting(data,loci=loci,control=contr)
mao.gc$npusr
mao.gc$npdat

## End(Not run)
```

---

**Kinship coefficient and genetic index of familiality**

**Description**

The genetic index of familiality is defined as the mean kinship between all pairs of individuals in a set multiplied by 100,000. Formally, it is defined as

\[
100,000 \times \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} k_{ij}
\]

where \( n \) is the number of individuals in the set and \( k_{ij} \) is the kinship coefficient between individuals \( i \) and \( j \).

The scaling is purely for convenience of presentation.

**Usage**

```r
gif(data, gifset)
```
Arguments

data the trio data of a pedigree
gifset a subgroup of pedigree members

Value

The returned value is a list containing:

gifval the genetic index of familiarity

References


Note

Adapted from gif.c, testable with -Dexecutable as standalone program, which can be use for any pair of individuals

Author(s)

Alun Thomas, Jing Hua Zhao

See Also

pfc

Examples

```r
## Not run:
test<-c(
  5, 0, 0,
  1, 0, 0,
  9, 5, 1,
  6, 0, 0,
  10, 9, 6,
  15, 9, 6,
  21, 10, 15,
  3, 0, 0,
  18, 3, 15,
  23, 21, 18,
  2, 0, 0,
  4, 0, 0,
  7, 0, 0,
  8, 4, 7,
  11, 5, 8,
  12, 9, 6,
  13, 9, 6,
  14, 5, 8,
  16, 14, 6,
```
h2

Heritability estimation according to twin correlations

Description
Heritability and variance estimation according to twin pair correlations.

Usage
h2(mzDat=NULL,dzDat=NULL,rmz=NULL,rdz=NULL,nmz=NULL,ndz=NULL,selV=NULL)

Arguments
mzDat a data frame for monzygotic twins (MZ)
dzDat a data frame for dizygotic twins (DZ)
rmz correlation for MZ twins
rdz correlation for DZ twins
nmz sample size for MZ twins
ndz sample size for DZ twins
selV names of variables for twin and cotwin

Details
The example section shows how to obtain bootstrap 95% CI.

Value
The returned value is a matrix containing heritability and their variance estimations for "h2","c2","e2","vh","vc","ve".

References
Keeping ES. Introduction to Statistical Inference, Dover Publications, Inc. 1995

Author(s)
Jing Hua Zhao
Examples

## Not run:

```r
ACE_CI <- function(mzData, dzData, n.sim=5, selV=NULL, verbose=TRUE) {
  ACE_twinData <- h2(mzDat=mzData, dzDat=dzData, selV=selV)
  print(ACE_twinData)
}

nmz <- dim(mzData)[1]
ndz <- dim(dzData)[1]
a <- vector()
set.seed(12345)
for(i in 1:n.sim) {
  cat("Running #", i,"/", n.sim,"\r",sep="\\n")
  sampled_mz <- sample(1:nmz, replace=TRUE)
  sampled_dz <- sample(1:ndz, replace=TRUE)
  mzDat <- mzData[sampled_mz]
  dzDat <- dzData[sampled_dz]
  ACE_i <- h2(mzDat=mzDat, dzDat=dzDat, selV=selV)
  if(verbose) print(ACE_i)
  a <- rbind(a, ACE_i)
}

heritability according to correlations

ar <- as.data.frame(a)
m <- mean(ar, na.rm=TRUE)
s <- sd(ar, na.rm=TRUE)
allr <- data.frame(mean=m, sd=s, lcl=m-1.96*s, ucl=m+1.96*s)
print(allr)
}

selVars <- c("bmi1", "bmi2")

library(mvtnorm)
n.sim <- 500
cat("The first study")
mzm <- as.data.frame(rmvnorm(195, c(22.75, 22.75),
  matrix(2.66^2*c(1, 0.67, 0.67, 1), 2)))
dzm <- as.data.frame(rmvnorm(130, c(23.44, 23.44),
  matrix(2.75^2*c(1, 0.32, 0.32, 1), 2)))
mzw <- as.data.frame(rmvnorm(384, c(21.44, 21.44),
  matrix(3.08^2*c(1, 0.72, 0.72, 1), 2)))
dzw <- as.data.frame(rmvnorm(243, c(21.72, 21.72),
  matrix(3.12^2*c(1, 0.33, 0.33, 1), 2)))

names(mzm) <- names(dzm) <- names(mzw) <- names(dzw) <- c("bmi1", "bmi2")
ACE_CI(mzm, dzm, n.sim, selV=selVars, verbose=FALSE)
ACE_CI(mzw, dzw, n.sim, selV=selVars, verbose=FALSE)
```

## End(Not run)
Description

Haplotype reconstruction using sorting and trimming algorithms

Usage

\[ \text{hap}(id, data, nloci, loci=\text{rep}(2, nloci), names=paste("loci", 1:nloci, sep=""), \]
\[ \quad \text{control}=\text{hap.control}()) \]

Arguments

- `id` a column of subject id
- `data` genotype table
- `nloci` number of loci
- `loci` number of alleles at all loci
- `names` locus names
- `control` is a function with the following arguments,
  1. `mb` Maximum dynamic storage to be allocated, in Mb
  2. `pr` Prior (ie population) probability threshold
  3. `po` Posterior probability threshold
  4. `to` Log-likelihood convergence tolerance
  5. `th` Posterior probability threshold for output
  6. `maxit` Maximum EM iteration
  7. `n` Force numeric allele coding (1/2) on output (off)
  8. `ss` Tab-delimited spreadsheet file output (off)
  9. `rs` Random starting points for each EM iteration (off)
 10. `rp` Restart from random prior probabilities
 11. `ro` Loci added in random order (off)
 12. `rv` Loci added in reverse order (off)
 13. `sd` Set seed for random number generator (use date+time)
 14. `mm` Repeat final maximization multiple times
 15. `mi` Create multiple imputed datasets. If set >0
 16. `mc` Number of MCMC steps between samples
 17. `ds` Starting value of Dirichlet prior parameter
 18. `de` Finishing value of Dirichlet prior parameter
 19. `q` Quiet operation (off)
 20. `hapfile` a file for haplotype frequencies
 21. `assignfile` a file for haplotype assignment
Details

The package can handle much larger number of multiallelic loci. For large sample size with relatively small number of multiallelic loci, genecounting should be used.

Value

The returned value is a list containing:

- log-likelihood assuming linkage disequilibrium
- convergence status, 0=failed, 1=succeeded
- number of iterations

References


Note

adapted from hap

See Also

genecounting

Examples

```r
## Not run:
# 4 SNP example, to generate hap.out and assign.out alone
data(fsnps)
hap(id=fsnps[,1],data=fsnps[,3:18],nloci=4)
dir()
file.show("hap.out")
file.show("assign.out")

# to generate results of imputations
control <- hap.control(ss=1,mi=5,hapfile="h",assignfile="a")
 hap(id=fsnps[,1],data=fsnps[,3:18],nloci=4,control=control)
dir()

## End(Not run)
```
hap.em

**Gene counting for haplotype analysis**

---

**Description**

Gene counting for haplotype analysis with missing data, adapted for hap.score

**Usage**

```
hap.em(id, data, locus.label=NA, converge.eps=1e-06, maxiter=500, miss.val=0)
```

**Arguments**

- `id` a vector of individual IDs
- `data` Matrix of alleles, such that each locus has a pair of adjacent columns of alleles, and the order of columns corresponds to the order of loci on a chromosome. If there are K loci, then ncol(data) = 2*K. Rows represent alleles for each subject.
- `locus.label` Vector of labels for loci, of length K (see definition of data matrix).
- `converge.eps` Convergence criterion, based on absolute change in log likelihood (lnlike).
- `maxiter` Maximum number of iterations of EM
- `miss.val` missing value

**Value**

List with components:

- `converge` Indicator of convergence of the EM algorithm (1=converged, 0 = failed).
- `niter` Number of iterations completed in the EM algorithm.
- `locus.info` A list with a component for each locus. Each component is also a list, and the items of a locus-specific list are the locus name and a vector for the unique alleles for the locus.
- `locus.label` Vector of labels for loci, of length K (see definition of input values).
- `haplotype` Matrix of unique haplotypes. Each row represents a unique haplotype, and the number of columns is the number of loci.
- `hap.prob` Vector of mle’s of haplotype probabilities. The ith element of hap.prob corresponds to the ith row of haplotype.
- `lnlike` Value of lnlike at last EM iteration (maximum lnlike if converged).
- `indx.subj` Vector for index of subjects, after expanding to all possible pairs of haplotypes for each person. If indx=i, then i is the ith row of input matrix data. If the ith subject has n possible pairs of haplotypes that correspond to their marker phenotype, then i is repeated n times.
- `nreps` Vector for the count of haplotype pairs that map to each subject’s marker genotypes.
hap1code  Vector of codes for each subject’s first haplotype. The values in hap1code are the row numbers of the unique haplotypes in the returned matrix haplotype.

hap2code  Similar to hap1code, but for each subject’s second haplotype.

post  Vector of posterior probabilities of pairs of haplotypes for a person, given their marker phenotypes.

References
See hap

Note
Adapted from HAP

Author(s)
Jing Hua Zhao

See Also
hap, LDkl

Examples

## Not run:
data(hla)
hap.em(id=1:length(hla[,1]),data=hla[,3:8],locus.label=c("DQR","DQA","DQB"))

## End(Not run)

---

hap.score  Score statistics for association of traits with haplotypes

Description
Compute score statistics to evaluate the association of a trait with haplotypes, when linkage phase is unknown and diploid marker phenotypes are observed among unrelated subjects. For now, only autosomal loci are considered. This package haplo.score which this function is based is greatly acknowledged.

Usage

```r
hap.score(y, geno, trait.type="gaussian", offset=NA, x.adj=NA,skip.haplo=0.005, locus.label=NA, miss.val=0, n.sim=0, method="gc", id=NA, handle.miss=0, mloci=NA, sexid=NA)
```
Arguments

- **y**: Vector of trait values. For trait.type = "binomial", y must have values of 1 for event, 0 for no event.
- **geno**: Matrix of alleles, such that each locus has a pair of adjacent columns of alleles, and the order of columns corresponds to the order of loci on a chromosome. If there are K loci, then ncol(geno) = 2*K. Rows represent alleles for each subject.
- **trait.type**: Character string defining type of trait, with values of "gaussian", "binomial", "poisson", "ordinal".
- **offset**: Vector of offset when trait.type = "poisson"
- **x.adj**: Matrix of non-genetic covariates used to adjust the score statistics. Note that intercept should not be included, as it will be added in this function.
- **skip.haplo**: Skip score statistics for haplotypes with frequencies < skip.haplo
- **locus.label**: Vector of labels for loci, of length K (see definition of geno matrix).
- **miss.val**: Vector of codes for missing values of alleles.
- **n.sim**: Number of simulations for empirical p-values. If n.sim=0, no empirical p-values are computed.
- **method**: method of haplotype frequency estimation, "gc" or "hap"
- **id**: an added option which contains the individual IDs
- **handle.miss**: flag to handle missing genotype data, 0=no, 1=yes
- **mloci**: maximum number of loci/sites with missing data to be allowed in the analysis
- **sexid**: flag to indicator sex for data from X chromosome, i=male, 2=female

Details

This is a version which substitutes haplo.em

Value

List with the following components:

- **score.global**: Global statistic to test association of trait with haplotypes that have frequencies \( \geq \) skip.haplo.
- **df**: Degrees of freedom for score.global.
- **score.global.p**: P-value of score.global based on chi-square distribution, with degrees of freedom equal to df.
- **score.global.p.sim**: P-value of score.global based on simulations (set equal to NA when n.sim=0).
- **score.haplo**: Vector of score statistics for individual haplotypes that have frequencies \( \geq \) skip.haplo.
- **score.haplo.p**: Vector of p-values for score.haplo, based on a chi-square distribution with 1 df.
- **score.haplo.p.sim**: Vector of p-values for score.haplo, based on simulations (set equal to NA when n.sim=0).
score.max.p.sim

P-value of maximum score.haplo, based on simulations (set equal to NA when n.sim=0).

haplotype

Matrix of haplotypes analyzed. The ith row of haplotype corresponds to the ith item of score.haplo, score.haplo.p, and score.haplo.p.sim.

hap.prob

Vector of haplotype probabilities, corresponding to the haplotypes in the matrix haplotype.

locus.label

Vector of labels for loci, of length K (same as input argument).

n.sim

Number of simulations.

n.val.global

Number of valid simulated global statistics.

n.val.haplo

Number of valid simulated score statistics (score.haplo) for individual haplotypes.

References

Schaid DJ, Rowland CM, Tines DE, Jacobson RM, Poland GA (2002) Score tests for association of traits with haplotypes when linkage phase is ambiguous. Amer J Hum Genet 70:425-34

Examples

```r
## Not run:
data(hla)
y<-hla[,2]
geno<-hla[,3:8]
# complete data
hap.score(y,geno,locus.label=c("DRB","DQA","DQB"))
# incomplete genotype data
hap.score(y,geno,locus.label=c("DRB","DQA","DQB"),handle.miss=1,mloci=1)
unlink("assign.dat")

### note the differences in p values in the following runs

data(aldh2)
# to subset the data since hap doesn't handle one allele missing
deleted<-c(40,239,256)
aldh2[deleted,]
aldh2<-aldh2[-deleted,]
y<-aldh2[,2]
geno<-aldh2[3:18]
# only one missing locus
hap.score(y,geno,handle.miss=1,mloci=1,method="hap")
# up to seven missing loci and with 10,000 permutations
hap.score(y,geno,handle.miss=1,mloci=7,method="hap",n.sim=10000)

# hap.score takes considerably longer time and does not handle missing data
hap.score(y,geno,n.sim=10000)

## End(Not run)
```
**hla**  
*The HLA data*

**Description**

This data set contains HLA markers DRB, DQA, DQB and phenotypes of 271 Schizophrenia patients (y=1) and controls (y=0). Genotypes for 3 HLA loci have prefixes name (e.g., "DQB") and a suffix for each of two alleles (".a1" and ".a2").

**Usage**

```r
data(hla)
```

**Format**

A data frame containing 271 rows and 8 columns

**Source**

Dr Padraig Wright of Pfizer

**htr**  
*Haplotype trend regression*

**Description**

Haplotype trend regression (with permutation)

**Usage**

```r
htr(y,x,n.sim=0)
```

**Arguments**

- `y`  
  A vector of phenotype
- `x`  
  A haplotype table
- `n.sim`  
  The number of permutations

**Value**

The returned value is a list containing:

- `f`  
  The F statistic for overall association
- `p`  
  The p value for overall association
- `fv`  
  The F statistics for individual haplotypes
- `pi`  
  The p values for individual haplotypes
References


Note

adapted from emgi.cpp, a pseudorandom number seed will be added on

Author(s)

Dimitri Zaykin, Jing Hua Zhao

See Also

hap.score

Examples

```r
### Not run:
# 26-10-03
# this is now part of demo
test2<-read.table("test2.dat")
y<-test2[,1]
x<-test2[,,-1]
y<-as.matrix(y)
x<-as.matrix(x)
htest2<-httr(y,x)
htest2
htest2<-httr(y,x,n.sim=10)
htest2

# 13-11-2003
data(apoeapoc)
apoeapoc.gc<-gc.em(apoeapoc[,5:8])
y<-apoeapoc$y
for(i in 1:length(y)) if(y[i]==2) y[i]<-1
htr(y,apoeapoc.gc$htrtable)

# 20-8-2008
# part of the example from useR!2008 tutorial by Andrea Foulkes
# It may be used beyond the generalized linear model (GLM) framework
HaploEM <- haplo.em(Geno,locus.label=SNPnames)
HapMat <- HapDesign(HaploEM)
m1 <- lm(Trait+HapMat)
m2 <- lm(Trait~1)
apova(m2,m1)

### End(Not run)
```
Hardy-Weinberg equilibrium test for a multiallelic marker

Description

Hardy-Weinberg equilibrium test

Usage

hwe(data, data.type="allele", yates.correct=FALSE, miss.val=0)

Arguments

data A rectangular data containing the genotype, or an array of genotype counts

data.type An option taking values "allele", "genotype", "count" if data is alleles, genotype or genotype count

yates.correct A flag indicating if Yates’ correction is used for Pearson $\chi^2$ statistic

miss.val A list of missing values

Details

This function obtains Hardy-Weinberg equilibrium test statistics. It can handle data coded as allele numbers (default), genotype identifiers (by setting data.type="genotype") and counts corresponding to individual genotypes (by setting data.type="count") which requires that genotype counts for all $n(n+1)$ possible genotypes, with $n$ being the number of alleles.

For highly polymorphic markers when asymptotic results do not hold, please resort to hwe.hardy.

Value

The returned value is a list containing:

- allele.freq Frequencies of alleles
- x2 Pearson $\chi^2$
- p.x2 p value for $\chi^2$
- lrt Log-likelihood ratio test statistic
- p.lrt p value for lrt
- df Degree(s) of freedom
- rho $\sqrt{\chi^2/N}$ the contingency table coefficient

Author(s)

Jing Hua Zhao

See Also

hwe.hardy
Examples

```r
## Not run:
a <- c(3, 2, 2)
a.out <- hwe(a, data.type="genotype")
a.out
a.out <- hwe(a, data.type="count")
a.out
require(haplo.stats)
data(hla)
hla.DQR <- hwe(hla[,3:4])
summary(hla.DQR)
# multiple markers
s <- vector()
for(i in seq(3,8,2))
{
  hwe_i <- hwe(hla[,i:(i+1)])
  s <- rbind(s,hwe_i)
}
s
## End(Not run)
```

---

**hwe.cc**  
*A likelihood ratio test of population Hardy-Weinberg equilibrium for case-control studies*

**Description**

A likelihood ratio test of population Hardy-Weinberg equilibrium for case-control studies

**Usage**

`hwe.cc(model, case, ctrl, k0, initial1, initial2)`

**Arguments**

- **model**: model specification, dominant, recessive
- **case**: a vector of genotype counts in cases
- **ctrl**: a vector of genotype counts in controls
- **k0**: prevalence of disease in the population
- **initial1**: initial values for beta, gamma, and q
- **initial2**: initial values for logit(p) and log(gamma)
Details

This is a collection of utility functions. The null hypothesis declares that the proportions of genotypes are according to Hardy-Weinberg law, while under the alternative hypothesis, the expected genotype counts are according to the probabilities that particular genotypes are obtained conditional on the prevalence of disease in the population. In so doing, Hardy-Weinberg equilibrium is considered using both case and control samples but pending on the disease model such that 2-parameter multiplicative model is built on baseline genotype $\alpha, \alpha \beta$ and $\alpha \gamma$.

Value

The returned value is a list with the following components.

- **Cox**: statistics under a general model
- **t2par**: under the null hypothesis
- **t3par**: under the alternative hypothesis
- **lrt.stat**: the log-likelihood ratio statistic
- **pval**: the corresponding p value

References


Author(s)

Chang Yu, [http://biostat.mc.vanderbilt.edu/wiki/Main/ChangYu](http://biostat.mc.vanderbilt.edu/wiki/Main/ChangYu), Li Wang, Jing Hua Zhao

See Also

- **hwe**

Examples

```r
### Not run:

#### Saba Sile, email of Jan 26, 2007, data always in order of GG AG AA, p=Pr(G),
#### q=1-p=Pr(A)
#### case=c(155,27,4)
#### ctrl=c(408,55,15)
#### k0=.2
#### initial1=c(1.0,0.94,0.0904)
#### initial2=c(logit(1-0.0904),log(0.94))
#### hwe.cc("recessive",case,ctrl,k0,initial1, initial2)

#### John Phillips III, TGFb1 data codon 10: TT CT CC, CC is abnormal and increasing
#### TGFb1 activity
#### case=c(29,78,13)
#### ctrl=c(17,28,6)
#### k0 <- 1e-5
#### initial1 <- c(2.45,2.45,0.34)
```
### Description

Hardy-Weinberg equilibrium test by MCMC

### Usage

```r
hwe.hardy(a, alleles = 3, seed = 3000, sample = c(1000, 1000, 5000))
```

### Arguments

- **a**: an array containing the genotype counts, as integer.
- **alleles**: number of allele at the locus, greater than or equal to 3, as integer.
- **seed**: pseudo-random number seed, as integer.
- **sample**: optional, parameters for MCMC containing number of chunks, size of a chunk and burn-in steps, as integer.

### Value

The returned value is a list containing:

- **method**: Hardy-Weinberg equilibrium test using MCMC
- **data.name**: name of used data if `x` is given
- **p.value**: Monte Carlo p value
- **p.value.se**: standard error of Monte Carlo p value
- **switches**: percentage of switches (partial, full and altogether)

### References


### Note

Adapted from HARDY, testable with -Dexecutable as standalone program

### Author(s)

Sun-Wei Guo, Jing Hua Zhao, Gregor Gorjanc
Source

http://www.stat.washington.edu/thompson/Genepi/pangaea.shtml,

See Also

hwe, HWE.test.genotype

Examples

```r
## Not run:
# example 2 from hwe.doc:
a <- c(3, 4, 2, 2, 2, 2, 3, 3, 2, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 1, 0, 0, 0)
ex2 <- hwe.hardy(a=a, alleles=8)

# example using HLA
data(hla)
x <- hla[,3:4]
y <- pgc(x,handle.miss=0,with.id=1)
n.alleles <- max(x,na.rm=TRUE)
z <- vector("numeric",n.alleles*(n.alleles+1)/2)
z[y$idsave] <- y$wt
hwe.hardy(a=z,alleles=n.alleles)

# with use of class 'genotype'
# this is to be fixed
library(genetics)
hlagen <- genotype(a1=x$DQR.a1, a2=x$DQR.a2,
allocated=sort(unique(c(x$DQR.a1, x$DQR.a2))))
hwe.hardy(hlagen)

# comparison with hwe
hwe(z, data.type="count")

# to create input file for HARDY
print.tri <- function (xx,n) {
cat(n, "\n")
  for (i in 1:n) {
    for (j in 1:i) {
      cat(xx[i,j], " "
    }
  cat("\n")
  }
  cat("100 170 1000\n")
}
```
kin.morgan

kinship matrix for simple pedigree

Description

kinship matrix according to Morgan v2.1

Usage

kin.morgan(ped, verbose=FALSE)

Arguments

ped individual’s id, father’s id and mother’s id
verbose an option to print out the original pedigree

Value

The returned value is a list containing:

kin the kinship matrix in vector form
kin.matrix the kinship matrix

References

Morgan V2.1 http://www.stat.washington.edu/thompson/Genepi/MORGAN/Morgan.shtml

Note

The input data is required to be sorted so that parents precede their children

Author(s)

Morgan development team, Jing Hua Zhao

See Also

gif
Examples

```r
## Not run:
# Werner syndrome pedigree
werner<-c(
  1, 0, 0, 1,
  2, 0, 0, 2,
  3, 0, 0, 2,
  4, 1, 2, 1,
  5, 0, 0, 1,
  6, 1, 2, 2,
  7, 1, 2, 2,
  8, 0, 0, 1,
  9, 4, 3, 2,
 10, 5, 6, 1,
 11, 5, 6, 2,
 12, 8, 7, 1,
 13,10, 9, 2,
 14,12, 11, 1,
 15,14, 13, 1)
werner<-t(matrix(werner,nrow=4))
kin.morgan(werner[,1:3])
```

## End(Not run)

**klem**  

*Haplotype frequency estimation based on a genotype table of two multiallelic markers*

**Description**

Haplotype frequency estimation using expectation-maximization algorithm based on a table of genotypes of two multiallelic markers.

**Usage**

`klem(obs, k=2, l=2)`

**Arguments**

- `obs` a table of genotype counts
- `k` number of alleles at marker 1
- `l` number of alleles at marker 2

**Details**

The dimension of the genotype table should be `k*(k+1)/2 x l*(l+1)/2`.  
Modified from 2ld.c.
Value
The returned value is a list containing:

- haplotype Frequencies
- log-likelihood under linkage equilibrium
- log-likelihood under linkage disequilibrium

Author(s)
Jing Hua Zhao

See Also
genecounting

Examples

```r
## Not run:
# an example with known genotype counts
z <- klem(obs=1:9)
# an example with imputed genotypes at SH2B1
cwd <- getwd()
cs.dir <- file.path(path.package("gap"),"tests/klem")
setwd(cs.dir)
dir()
source("SH2B1.R",echo=TRUE)
setwd(cwd)

## End(Not run)
```

Description
An example pedigree data

Usage
data(151)

Format
A data frame

Source
Morgan v3.
References


Examples

```r
## Not run:
k_m <- kin.morgan(151)
k2 <- km$kin.matrix*2

# quantitative trait
library(regress)
r <- regress(qt ~ 1, ~k2, data=151)
names(r)
r
# qualitative trait
N <- dim(151)[1]
w <- with(151,quantile(qt, probs=0.75, na.rm=TRUE))
ped51 <- within(151, bt <- ifelse(qt<w, 0, 1))
d <- regress(bt ~ 1, ~k2, data=ped51)
d
# for other tests not shown here
set.seed(12345)
ped51 <- within(ped51,(r <- rnorm(N); bt[is.na(bt)] <- 0))
library(foreign)
write.dta(ped51,"ped51.dta")
## End(Not run)
```

---

**LD22**

**LD statistics for two diallelic markers**

**Description**

LD statistics for two SNPs.

It is possible to perform permutation test of $r^2$ by re-ordering the genotype through R’s sample function, obtaining the haplotype frequencies by `gc.em` or `genecounting`, supplying the estimated haplotype frequencies to the current function and record x2, and comparing the observed x2 and that from the replicates.

**Usage**

`LD22(h,n)`

**Arguments**

- `h` a vector of haplotype frequencies
- `n` number of haplotypes
Value

The returned value is a list containing:

- `h`: the original haplotype frequency vector
- `n`: the number of haplotypes
- `D`: the linkage disequilibrium parameter
- `VarD`: the variance of D
- `Dmax`: the maximum of D
- `VarDmax`: the variance of Dmax
- `Dprime`: the scaled disequilibrium parameter
- `VarDprime`: the variance of Dprime
- `x2`: the Chi-squared statistic
- `lor`: the log(OR) statistic
- `vlor`: the var[log(OR)] statistic

References

Zabetian CP, Buxbaum SG, Elston RC, Kohnke MD, Anderson GM, Gelernter J, Cubells JF. The structure of linkage disequilibrium at the DBH locus strongly influences the magnitude of association between diallelic markers and plasma dopamine beta-hydroxylase activity Am J Hum Genet 72: 1389-1400


Note

extracted from 2ld.c

Author(s)

Jing Hua Zhao

See Also

LDk1

Examples

```r
# Not run:
h <- c(0.442356, 0.291532, 0.245794, 0.020319)
n <- 481*2
t <- LD22(h, n)
t
```

```r
## End(Not run)```
LDkl

LD statistics for two multiallelic markers

Description

LD statistics for two multiallelic loci. For two diallelic markers, the familiar $r^2$ has standard error $\text{seX}2$.

Usage

LDkl(n1=2,n2=2,h,n,optrho=2,verbose=FALSE)

Arguments

- n1: number of alleles at marker 1
- n2: number of alleles at marker 2
- h: a vector of haplotype frequencies
- n: number of haplotypes
- optrho: type of contingency table association, 0=Pearson, 1=Tschuprow, 2=Cramer (default)
- verbose: detailed output of individual statistics

Value

The returned value is a list containing:

- n1: the number of alleles at marker 1
- n2: the number of alleles at marker 2
- h: the haplotype frequency vector
- n: the number of haplotypes
- $D'$: $D'$
- VarDp: variance of $D'$
- Dijtable: table of Dij
- VarDijtable: table of variances for Dij
- Dmaxtable: table of Dmax
- Dijptable: table of Dij'
- VarDijptable: table of variances for Dij'
- X2table: table of Chi-squares (based on Dij)
- ptable: table of p values
- x2: the Chi-squared statistic
- seX2: the standard error of $x2/n$
LDkl

rho the measure of association
seR the standard error of rho
optrho the method for calculating rho
klinfo the Kullback-Leibler information

References


Note

adapted from 2ld.c

Author(s)

Jing Hua Zhao

See Also

LD22

Examples

## Not run:
# two examples in the C program 2LD:
# two SNPs as in 2by2.dat
# this can be compared with output from LD22

h <- c(0.442356,0.291532,0.245794,0.020319)
n <- 481*2
t <- LDkl(2,2,h,n)
t

# two multiallelic markers as in kbyl.dat
# the two-locus haplotype vector is in file "kbyl.dat"

filespec <- system.file("tests/2ld/kbyl.dat")
h <- scan(filespec,skip=1)
t <- LDkl(9,5,h,213*2,verbose=TRUE)

## End(Not run)
**Description**

A multi-generational pedigree containing individual, father, mother IDs and sex.

**Usage**

```r
data(lukas)
```

**Format**

An example pedigree

**Source**

Lukas Keller

---

**makeped**

*A function to prepare pedigrees in post-MAKEPED format*

---

**Description**

Many computer programs for genetic data analysis requires pedigree data to be in the so-called “post-MAKEPED” format. This function performs this translation and allows for some inconsistencies to be detected.

The first four columns of the input file contains the following information:

- Pedigree ID
- Individual ID
- Father’s ID
- Mother’s ID

Sex coding is 0=unknown, 1=male, 2=female. These can be followed by satellite information such as disease phenotype and marker information.

The output file has extra information extracted from data above.

**Usage**

```r
makeped(pifile="pedfile.pre", pofile="pedfile.ped", auto.select=1, with.loop=0, loop.file=NA, auto.proband=1, proband.file=NA)
```
Arguments

- **pifile**: input filename
- **pofile**: output filename
- **auto.select**: no loops in pedigrees and probands are selected automatically? 0=no, 1=yes
- **with.loop**: input data with loops? 0=no, 1=yes
- **loop.file**: filename containing pedigree id and an individual id for each loop, set if with.loop=1
- **auto.proband**: probands are selected automatically? 0=no, 1=yes
- **proband.file**: filename containing pedigree id and proband id, set if auto.proband=0 (not implemented)

Details

Before invoking makeped, input file, loop file and proband file have to be prepared.

By default, auto.select=1, so translation proceeds without considering loops and proband statuses. If there are loops in the pedigrees, then set auto.select=0, with.loop=1, loop.file="filespec".

There may be several versions of makeped available, but their differences with this port should be minor.

Value

All output will be written in pofile

Note

adapted from makeped.c by W Li and others

Source

http://linkage.rockefeller.edu

Examples

```r
## Not run:
cwd <- getwd()
cs.dir <- file.path(path.package("gap"),"tests","kinship")
setwd(cs.dir)
dir()
makeped("ped7.pre","ped7.ped",0,1,"ped7.lop")
setwd(cwd)

## End(Not run)
```
A study of Parkinson’s disease and MAO gene

Description

The markers are both with actual allele sizes and allele numbers. The dataset is distributed with the GENECOUNTING version 2.0 illustrating gene counting method involving chromosome X. A total of 183 patients and 157 controls (150 males, 190 females) were available, together with five markers in MAOA (monoamine oxidase A) region with alleles 12, 9, 6, 5, 3, and the first three markers were genotyped in all individuals while the fourth and fifth were genotyped for 294 and 304 individuals.

Usage

data(mao)

Format

A data frame

Source

Dr Helen Latsoudis of Institute of Psychiatry, KCL

References


Sample size calculation for mediation analysis

Description

The function computes sample size for regression problems where the goal is to assess mediation of the effects of a primary predictor by an intermediate variable or mediator.

Mediation has been thought of in terms of the proportion of effect explained, or the relative attenuation of $b_1$, the coefficient for the primary predictor $X_1$, when the mediator, $X_2$, is added to the model. The goal is to show that $b_1^*$, the coefficient for $X_1$ in the reduced model (i.e., the model with only $X_1$, differs from $b_1$, its coefficient in the full model (i.e., the model with both $X_1$ and the mediator $X_2$. If $X_1$ and $X_2$ are correlated, then showing that $b_2$, the coefficient for $X_2$, differs from zero is equivalent to showing $b_1^*$ differs from $b_1$. Thus the problem reduces to detecting an effect of $X_2$, controlling for $X_1$. In short, it amounts to the more familiar problem of inflating sample size to account for loss of precision due to adjustment for $X_1$. 
The approach here is to approximate the expected information matrix from the regression model including both \( X_1 \) and \( X_2 \), to obtain the expected standard error of the estimate of \( b_2 \), evaluated at the MLE. The sample size follows from comparing the Wald test statistic (i.e., the ratio of the estimate of \( b_2 \) to its SE) to the standard normal distribution, with the expected value of the numerator and denominator of the statistic computed under the alternative hypothesis. This reflects the Wald test for the statistical significance of a coefficient implemented in most regression packages.

The function provides methods to calculate sample sizes for the mediation problem for linear, logistic, Poisson, and Cox regression models in four cases for each model:

- \( \text{CpCm} \) continuous primary predictor, continuous mediator
- \( \text{BpCm} \) binary primary predictor, continuous mediator
- \( \text{CpBm} \) continuous primary predictor, binary mediator
- \( \text{BpBm} \) binary primary predictor, binary mediator

The function is also generally applicable to the analogous problem of calculating sample size adequate to detect the effect of a primary predictor in the presence of confounding. Simply treat \( X_2 \) as the primary predictor and consider \( X_1 \) the confounder.

### Usage

```r
masize(model, opts, alpha=0.025, gamma=0.2)
```

### Arguments

- **model**: "lineari", "logisticj", "poissonk", "coxl", where i,j,k,l range from 1 to 4,5,9,9, respectively.
- **opts**: A list specific to the model
  - \( b_1 \): regression coefficient for the primary predictor \( X_1 \)
  - \( b_2 \): regression coefficient for the mediator \( X_2 \)
  - \( \rho \): correlation between \( X_1 \) and \( X_2 \)
  - \( sdx_1, sdx_2 \): standard deviations (SDs) of \( X_1 \) and \( X_2 \)
  - \( f_1, f_2 \): prevalence of binary \( X_1 \) and \( X_2 \)
  - \( sdy \): residual SD of the outcome for the linear model
  - \( p \): marginal prevalence of the binary outcome in the logistic model
  - \( m \): marginal mean of the count outcome in a Poisson model
  - \( f \): proportion of uncensored observations for the Cox model
  - \( fc \): proportion of observations censored early
  - \( \alpha \): one-sided type-I error rate
  - \( \gamma \): type-II error rate
  - \( ns \): number of observations to be simulated
  - \( seed \): random number seed

For linear model, the arguments are \( b_2, \rho, sdx_2, sdy, \alpha, \) and \( \gamma \). For cases \( \text{CpBm} \) and \( \text{BpBm} \), set \( sdx_2 = \sqrt{f_2(1 - f_2)} \). Three alternative functions are included for the linear model. These functions make it possible to supply other combinations of input parameters affecting mediation:
b1* coefficient for the primary predictor
in the reduced model excluding the mediator (b1star)

b1 coefficient for the primary predictor
in the full model including the mediator

PTE proportion of the effect of the primary predictor
explained by the mediator, defined as (b1*-b1)/b1*

These alternative functions for the linear model require specification of an extra parameter, but are provided for convenience, along with two utility files for computing PTE and b1* from the other parameters. The required arguments are explained in comments within the R code.

alpha Type-I error rate, one-sided
gamma Type-II error rate

Details

For linear model, a single function, linear, implements the analytic solution for all four cases, based on Hsieh et al., is to inflate sample size by a variance inflation factor, \(1/(1-\rho^2)\), where rho is the correlation of X1 and X2. This also turns out to be the analytic solution in cases CpCm and BpCm for the Poisson model, and underlies approximate solutions for the logistic and Cox models. An analytic solution is also given for cases CpBm and BpBm for the Poisson model. Since analytic solutions are not available for the logistic and Cox models, a simulation approach is used to obtain the expected information matrix instead.

For logistic model, the approximate solution due to Hsieh is implemented in the function logistic.approx, and can be used for all four cases. Arguments are p, b2, rho, sdx2, alpha, and gamma. For a binary mediator with prevalence f2, sdx2 should be reset to \(\sqrt{f2(1-f2)}\). Simulating the information matrix of the logistic model provides somewhat more accurate sample size estimates than the Hsieh approximation. The functions for cases CpCm, BpCm, CpBm, and BpBm are respectively logistic.ccs, logistic.bcs, logistic.cbs, and logistic.bbs, as for the Poisson and Cox models. Arguments for these functions include p, b1, sdx1 or f1, b2, sdx2 or f2, rho, alpha, gamma, and ns. As in other functions, sdx1, sdx2, alpha, and gamma are set to the defaults listed above. These four functions call two utility functions, getb0 (to calculate the intercept parameter from the others) and antilogit, which are supplied.

For Poisson model, The function implementing the approximate solution based on the variance inflation factor is poisson.approx, and can be used for all four cases. Arguments are EY (the marginal mean of the Poisson outcome), b2, sdx2, rho, alpha and gamma, with sdx2 set to the usual defaults; use sdx2=\(\sqrt{f2(1-f2)}\) for a binary mediator with prevalence f2 (cases CpBm and BpBm). For cases CpCm and BpCm (continuous mediators), the approximate formula is also the analytic solution. For these cases, we supply redundant functions poisson.cc and poisson.bc, with the same arguments and defaults as for poisson.approx (it’s the same function). For the two cases with binary mediators, the functions are poisson.cb and poisson.bb. In addition to m, b2, f2, rho, alpha, and gamma, b1 and sdx1 or f1 must be specified. Defaults are as usual. Functions using simulation for the Poisson model are available: poisson.ccs, poisson.bcs, poisson.cbs, and poisson.bbs. As in the logistic case, these require arguments b1 and sdx1 or f1. For this case, however, the analytic functions are faster, avoid simulation error, and should be used. We include these functions as templates that could be adapted to other joint predictor distributions.
For Cox model, the function implementing the approximate solution, using the variance inflation factor and derived by Schmoor et al., is cox.approx, and can be used for all four cases. Arguments are \( b_2, \) \( sdx_2, \) \( \rho, \) \( \alpha, \) \( \gamma, \) and \( f. \) For binary \( X_2 \) set \( sdx_2 = \sqrt{f^2(1 - f^2)} \). The approximation works very well for cases CpCm and BpCm (continuous mediators), but is a bit less accurate for cases CpBm and BpBm (binary mediators). We get some improvement for those cases using the simulation approach. This approach is implemented for all four, as functions cox.ccs, cox.bcs, cox.cbs, and cox.bbs. Arguments are \( b_1, sdx_1 \) or \( f_1, b_2, sdx_2 \) or \( f_2, \rho, \alpha, \gamma, f, \) and \( ns, \) with defaults as described above. Slight variants of these functions, cox.ccs2, cox.bcs2, cox.cbs2, and cox.bbs2, make it possible to allow for early censoring of a fraction \( fc \) of observations; but in our experience this has virtually no effect, even with values of \( fc \) of 0.5. The default for \( fc \) is 0.

A summary of the argumentss is as follows, noting that additional parameter seed can be supplied for simulation-based method.

<table>
<thead>
<tr>
<th>model</th>
<th>arguments</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear1</td>
<td>( b_2, \rho, sdx_2, sdy )</td>
<td>linear</td>
</tr>
<tr>
<td>linear2</td>
<td>( b_{1,\text{star}}, PTE, \rho, sdx_1, sdy )</td>
<td>lineara</td>
</tr>
<tr>
<td>linear3</td>
<td>( b_{1,\text{star}}, b_2, PTE, sdx_1, sdx_2, sdy )</td>
<td>linearb</td>
</tr>
<tr>
<td>linear4</td>
<td>( b_{1,\text{star}}, b_1, b_2, sdx_1, sdx_2, sdy )</td>
<td>linearc</td>
</tr>
<tr>
<td>logistic1</td>
<td>( p, b_2, \rho, sdx_2 )</td>
<td>logistic.approx</td>
</tr>
<tr>
<td>logistic2</td>
<td>( p, b_1, b_2, \rho, sdx_1, sdx_2, ns )</td>
<td>logistic.ccs</td>
</tr>
<tr>
<td>logistic3</td>
<td>( p, b_1, f_1, b_2, \rho, sdx_2, ns )</td>
<td>logistic.bcs</td>
</tr>
<tr>
<td>logistic4</td>
<td>( p, b_1, b_2, f_2, \rho, sdx_1, ns )</td>
<td>logistic.cbs</td>
</tr>
<tr>
<td>logistic5</td>
<td>( p, b_1, f_1, b_2, f_2, \rho, ns )</td>
<td>logistic.bbs</td>
</tr>
<tr>
<td>poisson1</td>
<td>( m, b_2, \rho, sdx_2 )</td>
<td>poisson.approx</td>
</tr>
<tr>
<td>poisson2</td>
<td>( m, b_2, \rho, sdx_2 )</td>
<td>poisson.cc</td>
</tr>
<tr>
<td>poisson3</td>
<td>( m, b_2, \rho, sdx_2 )</td>
<td>poisson.bc</td>
</tr>
<tr>
<td>poisson4</td>
<td>( m, b_1, b_2, f_2, \rho, sdx_1 )</td>
<td>poisson.cb</td>
</tr>
<tr>
<td>poisson5</td>
<td>( m, b_1, f_1, b_2, f_2, \rho )</td>
<td>poisson.bb</td>
</tr>
<tr>
<td>poisson6</td>
<td>( m, b_1, b_2, \rho, sdx_1, sdx_2, ns )</td>
<td>poisson.ccs</td>
</tr>
<tr>
<td>poisson7</td>
<td>( m, b_1, f_1, b_2, \rho, sdx_2, ns )</td>
<td>poisson.bcs</td>
</tr>
<tr>
<td>poisson8</td>
<td>( m, b_1, b_2, f_2, \rho, sdx_1, ns )</td>
<td>poisson.cbs</td>
</tr>
<tr>
<td>poisson9</td>
<td>( m, b_1, f_1, b_2, f_2, \rho, ns )</td>
<td>poisson.bbs</td>
</tr>
<tr>
<td>cox1</td>
<td>( b_2, \rho, f, sdx_2 )</td>
<td>cox.approx</td>
</tr>
<tr>
<td>cox2</td>
<td>( b_1, b_2, \rho, f, sdx_1, sdx_2, ns )</td>
<td>cox.ccs</td>
</tr>
<tr>
<td>cox3</td>
<td>( b_1, f_1, b_2, \rho, f, sdx_2, ns )</td>
<td>cox.bcs</td>
</tr>
<tr>
<td>cox4</td>
<td>( b_1, b_2, f_2, \rho, f, sdx_1, ns )</td>
<td>cox.cbs</td>
</tr>
<tr>
<td>cox5</td>
<td>( b_1, f_1, b_2, f_2, \rho, f, ns )</td>
<td>cox.bbs</td>
</tr>
<tr>
<td>cox6</td>
<td>( b_1, b_2, \rho, f, f, sdx_1, sdx_2, ns )</td>
<td>cox.ccs2</td>
</tr>
<tr>
<td>cox7</td>
<td>( b_1, f_1, b_2, \rho, f, f, sdx_2, ns )</td>
<td>cox.bcs2</td>
</tr>
<tr>
<td>cox8</td>
<td>( b_1, b_2, f_2, \rho, f, f, sdx_1, ns )</td>
<td>cox.cbs2</td>
</tr>
<tr>
<td>cox9</td>
<td>( b_1, f_1, b_2, f_2, \rho, f, f, ns )</td>
<td>cox.bbs2</td>
</tr>
</tbody>
</table>
### Value

A short description of model (desc, b=binary, c=continuous, s=simulation) and sample size (n). In the case of Cox model, number of events (d) is also indicated.

### References


### Source

http://www.epibiostat.ucsf.edu/biostat/mediation/

### See Also

ab

### Examples

```r
## Not run:
## linear model
# CpGm
opts <- list(b2=0.5, rho=0.3, sdx2=1, sdy=1)
masize("linear1",opts)
# BpBm
opts <- list(b2=0.75, rho=0.3, f2=0.25, sdx2=sqrt(0.25*0.75), sdy=3)
masize("linear1",opts,gamma=0.1)

## logistic model
# CpGm
opts <- list(p=0.25, b2=log(0.5), rho=0.5, sdx2=0.5)
masize("logistic1",opts)
opts <- list(p=0.25, b1=log(1.5), sdx1=1, b2=log(0.5), f2=0.5, rho=0.5, ns=10000, seed=1234)
masize("logistic4",opts)
opts <- list(p=0.25, b1=log(1.5), sdx1=1, b2=log(0.5), f2=0.5, rho=0.5, ns=10000, seed=1234)
masize("logistic4",opts)

## Poisson model
# BpBm
opts <- list(m=0.5, b2=log(1.25), rho=0.3, sdx2=sqrt(0.25*0.75))
masize("poisson1",opts)
```
MCMCgrm

Mixed modeling with genetic relationship matrices

Description

Mixed modeling with genomic relationship matrix. This is appropriate with relationship matrix derived from family structures or unrelated individuals based on whole genome data.

Usage

MCMCgrm(model, prior, data, GRM, eps=0, n.thin=10, n.burnin=3000, n.iter=13000, ...)

Arguments

- **model**: statistical model
- **prior**: a list of priors for parameters in the model above
- **data**: a data.frame containing outcome and covariates
- **GRM**: a relationship matrix
- **eps**: a small number added to the diagonal of the a nonpositive definite GRM
- **n.thin**: thinning parameter in the MCMC
- **n.burnin**: the number of burn-in’s
- **n.iter**: the number of iterations
- **...**: other options as appropriate for MCMCglmm

Details

The function was created to address a number of issues involving mixed modelling with family data or population sample with whole genome data. First, the implementaiton will shed light on the uncertainty involved with polygenic effect in that posterior distributions can be obtained. Second, while the model can be used with the MCMCglmm package there is often issues with the specification of pedigree structures but this is less of a problem with genetic relationship matrices. We
can use established algorithms to generate kinship or genomic relationship matrix as input to the MCMCglmm function. Third, it is more intuitive to specify function arguments in line with other packages such as R2OpenBUGS, R2jags or glmmBUGS. In addition, our experiences of tuning the model would help to reset the input and default values.

Value

The returned value is an object as generated by MCMCglmm.

References


Author(s)

Jing Hua Zhao

Examples

```r
## Not run:
## with kinship

# library(kinship)
# fam <- with(l151,makefamid(id,fid,mid))
# s <-with(l151, makekinship(Fam, id, fid, mid))
# K <- as.matrix(s)*2

## with gap

s <- kin.morgan(l151)
K <- with(s,kin.matrix*2)
prior <- list(R=list(V=1, nu=0.002), G=list(G1=list(V=1, nu=0.002)))
m <- MCMCgmrm(qt~1,prior,l151,K)
save(m,file="l151.m")
pdf("l151.pdf")
plot(m)
dev.off()

## End(Not run)
```
Usage

```
metap(data, N, verbose="Y", prefixp="p", prefixn="n")
```

Arguments

- `data` : data frame
- `N` : Number of studies
- `verbose` : Control of detailed output
- `prefixp` : Prefix of p value, with default value "p"
- `prefixn` : Prefix of sample size, with default value "n"

Value

- `x2` : Fisher’s chi-squared statistics
- `p` : P values from Fisher’s method according to chi-squared distribution with 2*N degree(s) of freedom
- `z` : Combined z value
- `p1` : One-sided p value
- `p2` : Two-sided p value

Author(s)

Jing Hua Zhao

See Also

- `metareg`

Examples

```r
# Not run:
s <- data.frame(p1=0.1*rep(c(8:2),each=7),times=1),n1=rep(32000,49),
p2=0.1*rep(c(8:2),each=1),times=7),n2=rep(8000,49))
cbind(s,metap(s,2))

# Speilotes, Elizabeth K., M.D. [ESPIELOTES@PARTNERS.ORG]
# 22-2-2008 MRC-Epid JHZ

np <- 7
p <- 0.1^(rep(c(8:2),each=7)
qnorm((1-p)/2)
q <- c(32000,8000)
n1 <- n[1]
s1 <- s2 <- vector("numeric")

for (i in 1:np)
{

```
metareg

Fixed and random effects model for meta-analysis

Description

Given \( k = n \) studies with \( b_1, \ldots, b_N \) being \( \beta \)'s and \( se_1, \ldots, se_N \) standard errors from regression, the fixed effects model uses inverse variance weighting such that \( w_1 = 1/se_1^2, \ldots, w_N = 1/se_N^2 \) and the combined \( \beta \) as the weighted average, \( \beta_f = (b_1 * w_1 + \ldots + b_N * w_N)/w \), with \( w = w_1 + \ldots + w_N \) being the total weight, the se for this estimate is \( se_f = \sqrt{1/w} \).

For the random effects model, denote \( q_w = w_1 * (b_1 - \beta_f)^2 + \ldots + w_N * (b_N - \beta_f)^2 \) and \( dl = max(0, (q_w - (k - 1))/(w - (w_1^2 + \ldots + w_N^2)/w)) \), corrected weights are obtained such that \( w_1c = 1/(1/w_1 + dl), \ldots, w_Nc = 1/(1/w_N + dl) \), totaling \( w_c = w_1c + \ldots + w_Nc \). The combined \( \beta \) and se are then \( \beta_r = (b_1 * w_1c + \ldots + b_N * w_Nc)/w_c \) and \( se_r = \sqrt{1/w_c} \), leading to a z-statistic \( z_r = \beta_r/se_r \) and a p-value \( p_r = 2 * pnorm(-abs(z_r)) \). Moreover, a p-value testing for heterogeneity is \( p_{heter} = pchisq(q_w, k - 1, lower.tail = FALSE) \).

Usage

```
metareg(data, N, verbose="Y", prefixb="b", prefixse="se")
```
Arguments

- data : Data frame to be used
- N : Number of studies
- verbose : A control for screen output
- prefixb : Prefix of estimate; default value is "b"
- prefixse : Prefix of standard error; default value is "se" The function accepts a wide format data with estimates as $b_1, ..., b_N$ and standard errors as $se_1, ..., se_N$. More generally, they can be specified by prefixes in the function argument.

Value

The returned value is a data frame with the following variables:

- $p_f$ : P value (fixed effects model)
- $p_r$ : P value (random effects model)
- beta_f : regression coefficient
- beta_r : regression coefficient
- se_f : standard error
- se_r : standard error
- $z_f$ : $z$ value
- $z_r$ : $z$ value
- $p_{heter}$ : heterogeneity test p value
- i2 : $I^2$ statistic
- k : No of tests used
- eps : smallest double-precision number

References


Note

Adapted from a SAS macro

Author(s)

Shengxu Li, Jing Hua Zhao
A pedigree data on 282 animals deriving from two generations

Description


“The pedigrees for each of these 282 animals derive from an additional 24 base population (Generation 0) animals that do not have records of their own but, nevertheless, are of interest with respect to the inference on their own additive genetic values. Furthermore, it is presumed that these original 24 base animals are not related to each other. Therefore, the row dimension of u is 306 (282+24).” (Templeman & Rosa 2004)

Usage

data(meyer)

Format

A data frame containing 306 records

Source


Examples

```r
## Not run:
loutcome(library(gap)
meauty <- within(meyer,
  g1 <- ifelse(generation==1,1,0)
  g2 <- ifelse(generation==2,1,0)
))
lm(y~1+g1+g2,data=meyer)
```
library(MCMCglmm)
m <- MCMCglmm(y~1+g1+g2,random=animal~1,pedigree=meyer[,1:3],data=meyer,verbose=FALSE)
summary(m)
plot(m)

meyer <- within(meyer,
  id <- animal
  animal <- ifelse(!is.na(animal), animal, 0)
  dam <- ifelse(!is.na(dam), dam, 0)
  sire <- ifelse(!is.na(sire), sire, 0)
)
# library(kinship)
# A <- with(meyer,kinship(animal,sire,dam))%*%2
A <- kin.morgan(meyer)$kin.matrix%*%2

library(regress)
regress(y~1+g1+g2,-A,data=meyer)
prior <- list(R=list(V=1, nu=0.002), G=list(G1=list(V=1, nu=0.002)))
m2 <- MCMCgrm(y~1+g1+g2,prior,meyer,A,singular.ok=TRUE,verbose=FALSE)
summary(m2)
plot(m2)

### End(Not run)

---

**mfblong**

**Example data for ACEnucfam**

**Description**

This is the companion data for ACEnucfam.

**Usage**

data(mfblong)

**Format**

The data is a random subset of the birth weight data from the mental health registry of Norway.

- **male**: a dummy variable for being male; first-a dummy variable for being the first child; midage-a dummy variable for mother aged 20-35 at time of birth; highage-a dummy variable for mother older than 35 at time of birth and birthyr-year of birth minus 1967 (earliest birth year in birth registry).

**Source**

The data were obtained from the Biometrics website and preprocessed with f.mfb.R.
References


Description

To generate Manhattan plot, e.g., of genomewide significance (p values) and a random variable that is uniformly distributed. By default, a log10-transformation is applied. Note that with real chromosomal positions, it is also appropriate to plot and some but not all chromosomes. It is possible to specify options such as xlab and ylim when the plot is requested for data in other context.

Usage

mhtplot(data, control=mht.control(), hcontrol=hmht.control(), ...)

Arguments

data
A data frame with three columns representing chromosome, position and p values

control
A control function named mht.control() with the following arguments,

1. type. a flag with value "p" or "l" indicating if points or lines are to be drawn.
2. usepos. a flag to use real chromosomal positions as composed to ordinal positions with default value FALSE
3. logscale. a flag to indicate if p value is to be log-transformed with default value TRUE
4. base. the base of the logarithm with default value 10
5. cutoffs. the cut-offs where horizontal line(s) are drawn with default value NULL
6. colors. the color for different chromosome(s), and random if unspecified with default values NULL
7. labels. labels for the ticks on x-axis with default value NULL
8. srt. degree to which labels are rotated with default value of 45
9. gap. gap between chromosomes with default value NULL
10. cex. cex for the data points
11. yline. Margin line position
12. xline. Margin line position

hcontrol
A control function named hmht.control() with the following arguments,

1. data. chunk of data to be highlighted with default value NULL
2. colors. colors for annotated genes
3. yoffset. offset above the data point showing most significant p value with default value 0.5
4. cex. shrinkage factor for data points with default value 1.5
5. boxed. if the label for the highlighted region with default value FALSE
other options in compatible with the R plot function

Value

The plot is shown on or saved to the appropriate device.

Author(s)

Jing Hua Zhao

See Also

qqunif

Examples

```r
## Not run:
# foo example
test <- matrix(c(1,1,4,1,1,6,1,10,3,2,1,5,2,6,2,4,8),byrow=TRUE,6)
mhtplot(test)
mhtplot(test,mht.control(logscale=FALSE))

# fake example with Affy500k data
affy <-c(40220, 41400, 33801, 32334, 32056, 31470, 25835, 27457, 22864, 28501, 26273,
        24954, 19188, 15721, 14356, 15309, 11281, 14881, 6399, 12400, 7125, 6207)
CM <- cumsum(affy)
n.markers <- sum(affy)
n.chr <- length(affy)
test <- data.frame(chr=rep(1:n.chr,affy),pos=1:n.markers,p=runif(n.markers))

# to reduce size of the plot
# bitmap("mhtplot.bmp",res=72*5)
oldpar <- par()
par(cex=0.6)
colors <- rep(c("blue","green"),11)
# other colors, e.g.
# colors <- c("red","blue","green","cyan","yellow","gray","magenta","red","blue","green",
# "cyan","yellow","gray","magenta","red","blue","green","cyan","yellow","gray",
# "magenta","red")
mhtplot(test,control=mht.control(colors=colors),pch=19,srt=0)
title("A simulated example according to EPIC-Norfolk QCed SNPs")
axis(2)
axis(1,pos=0,labels=FALSE,tick=FALSE)
abline(0,0)
# dev.off()
par(oldpar)
```
mhtplot2

Manhattan plot with annotations

Description

To generate Manhattan plot with annotations. The function is generic and for instance could be used for genomewide p values or any random variable that is uniformly distributed. By default, a log10-transformation is applied. Note that with real chromosomal positions, it is also appropriate to plot and some but not all chromosomes.

It is possible to specify options such as xlab, ylim and font family when the plot is requested for data in other context.

To maintain back compatibility options as in mhtplot are used. The positions of the horizontal labels are now in the middle rather than at the beginning of their bands in the plot.
Usage
mhtplot2(data, control=mht.control(), hcontrol=hmht.control(), ...)

Arguments

data a data frame with three columns representing chromosome, position and p values
control A control function named mht.control() with the following arguments,
  1. type. a flag with value "p" or "l" indicating if points or lines are to be drawn.
  2. usepos. a flag to use real chromosomal positions as composed to ordinal positions with default value FALSE
  3. logscale. a flag to indicate if p value is to be log-transformed with default value TRUE
  4. base. the base of the logarithm with default value 10
  5. cutoffs. the cut-offs where horizontal line(s) are drawn with default value NULL
  6. colors. the color for different chromosome(s), and random if unspecified with default values NULL
  7. labels. labels for the ticks on x-axis with default value NULL
  8. srt. degree to which labels are rotated with default value of 45
  9. gap. gap between chromosomes with default value NULL
  10. cex. cex for the data points
  11. yline. Margin line position
  12. xline. Margin line position

hcontrol A control function named hmht.control() with the following arguments,
  1. data. chunk of data to be highlighted with default value NULL
  2. colors. colors for annotated genes
  3. yoffset. offset above the data point showing most significant p value with default value 0.5
  4. cex. shrinkage factor for data points with default value 1.5
  5. boxed. if the label for the highlighted region with default value FALSE

... other options in compatible with the R plot function

Value
The plot is shown on or saved to the appropriate device.

References

Author(s)
Jing Hua Zhao
Examples

## Not run:

# The following example uses only chromosomes 14 and 20 of the Nat Genet paper.

```r
mdata <- within(hr1420, {
  c1 <- colour = 1
  c2 <- colour = 2
  c3 <- colour = 3
  colour[c1] <- 62
  colour[c2] <- 73
  colour[c3] <- 552
})
mdata <- mdata[, c("CHR", "POS", "P", "gene", "colour")]
ops <- mht.control(colors = rep(c("lightgray", "gray"), 11), yline = 1.5, xline = 2, srt = 0)
hops <- hmht.control(data = subset(mdata, !is.na(gene)))
v <- "Verdana"
ifelse(Sys.info()["sysname"] == "Windows", windowsFonts(family = windowsFont(v)),
  family = v)
tiff("mht.tiff", width = .03937*189, height = .03937*189/2, units = "in", res = 1200,
  compress = "lzw")
par(las = 2, xpd = TRUE, cex.axis = 1.8, cex = 0.4)
mhtplot2(with(mdata, cbind(CHR, POS, P, colour)), hops, pch = 19,
  ylab = expression(paste(plain("-"), log(10), plain("p-value"), sep = "")),
  family = "family")
axis(2, pos = 2, at = seq(0, 25, 5), family = "family", cex = 0.5, cex.axis = 1.1)
dev.off()
```

## End(Not run)

---

**mia**

*multiple imputation analysis for hap*

---

**Description**

This command reads outputs from hap session that uses multiple imputations, i.e. -mi\# option. To simplify matters it assumes -ss option is specified together with -mi option there.

This is a very naive version of MIANALYZE, but can produce results for PROC MIANALYZE of SAS.
Usage

mia(hapfile, assfile, miafile, so, ns, mi, allsnps, sas)

Arguments

- **hapfile**: hap haplotype output file name
- **assfile**: hap assignment output file name
- **miafile**: mia output file name
- **so**: to generate results according to subject order
- **ns**: do not sort in subject order
- **mi**: number of multiple imputations used in hap
- **allsnps**: all loci are SNPs
- **sas**: produce SAS data step program

Details

It simply extracts outputs from hap

Value

The returned value is a list containing:

References


Note

adapted from hap, in fact cline.c and cline.h are not used

See Also

- hap

Examples

```r
## Not run:
# 4 SNP example, to generate hap.out and assign.out alone
data(fsnps)
hap(id=fsnps[,1],data=fsnps[,3:10],nloci=4)

# to generate results of imputations
ccontrol <- hap.control(ss=1,mi=5)
hap(id=fsnps[,1],data=fsnps[,3:10],nloci=4,control=control)

# to extract information from the second run above
```
Transmission/disequilibrium test of a multiallelic marker

Description

This function calculates transmission-disequilibrium statistics involving multiallelic marker.
Inside the function are tril and triu used to obtain lower and upper triangular matrices.

Usage

mtdt(x, n.sim=0)

Arguments

x the data table
n.sim the number of simulations

Value

It returned list contains the following components:

SE  Spielman-Ewens Chi-square from the observed data
ST  Stuart or score Statistic from the observed data
pSE the simulated p value
sSE standard error of the simulated p value
pST the simulated p value
sST standard error of the simulated p value
References


Author(s)

Mike Miller, Jing Hua Zhao

See Also

bt

Examples

```r
# Not run:

x <- matrix(c(0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,1,3,0,0,0,2,3,0,0,0,2,3,26,35,7,0,2,10,11,3,4,1,2,3,22,26,6,2,4,4,10,2,2,0,0,1,7,10,2,0,0,2,2,1,1,0,0,1,4,0,1,0,1,0,0,0,0,0,2,5,4,1,1,0,0,0,2,0,0,0,0,0,2,6,1,0,2,0,2,0,0,0,0,0,3,6,19,6,0,0,2,5,3,0,0,0,0,3,1,1,0,0,1,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0),nrow=12)

# See note to bt for the score test obtained by SAS

mtdt(x)

## End(Not run)
```

mtdt2

*Transmission/disequilibrium test of a multiallelic marker by Bradley-Terry model*
Description

This function calculates transmission-disequilibrium statistics involving multiallelic marker according to Bradley-Terry model.

Usage

\texttt{mtdt2(x, verbose=TRUE, n.sim=NULL, \ldots)}

Arguments

- \texttt{x}: the data table
- \texttt{verbose}: To print out test statistics if \texttt{TRUE}
- \texttt{n.sim}: Number of simulations
- \texttt{\ldots}: other options compatible with the \texttt{BTm} function

Value

It returned list contains the following components:

- \texttt{c2b}: A data frame in four-column format showing transmitted vs nontransmitted counts
- \texttt{BTm}: A fitted Bradley-Terry model object
- \texttt{X2}: Allele-wise, genotype-wise and goodness-of-fit Chi-squared statistics
- \texttt{df}: Degrees of freedom
- \texttt{p}: P value
- \texttt{pn}: Monte Carlo p values when \texttt{n.sim} is specified

References


Author(s)

Jing Hua Zhao

See Also

\texttt{mtdt}
Examples

### Not run:

```r
x <- matrix(c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              2, 3, 26, 35, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              2, 3, 22, 26, 6, 2, 4, 4, 10, 2, 2, 0, 0, 0,
              0, 1, 7, 10, 2, 0, 0, 2, 2, 1, 1, 0, 0, 0,
              0, 0, 1, 4, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
              0, 2, 5, 4, 1, 1, 0, 0, 0, 2, 0, 0, 0, 0,
              0, 0, 2, 6, 1, 0, 2, 0, 2, 0, 0, 0, 0, 0,
              0, 3, 6, 19, 6, 0, 0, 2, 5, 3, 0, 0, 0, 0,
              0, 0, 3, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), nrow=12)
xx <- mtdt2(x, refcat="12")
```

### End(Not run)

---

| muvar | Means and variances under 1- and 2-locus (biallelic) QTL model |

### Description

Function `muvar()` gives means and variances under 1-locus and 2-locus QTL model (simple); in the latter case it gives results from different avenues. This function is included for experimental purpose and yet to be generalized.

### Arguments

- `n.loci`: number of loci, 1=single locus, 2=two loci
- `y1`: the genotypic means of aa, Aa and AA
- `p1`: the frequency of the lower allele, or the that for the first locus under a 2-locus model
- `y12`: the genotypic means of aa, Aa and AA at the first locus and bb, Bb and BB at the second locus
- `p2`: the frequency of the lower allele at the second locus

### Details

```r
muvar(n.loci=1, y1=c(0,1,1), p1=0.5)
muvar(n.loci=2, y12=c(1,1,1,1,0,0,0,0), p1=0.99, p2=0.9)
```
mvmeta

Value

Currently it does not return any value except screen output; the results can be kept via R’s sink() command or via modifying the C/R codes.

References


Note

Adapted from an earlier C program written for the above book

Author(s)

Jing Hua Zhao

Examples

```r
## Not run:
# the default 1-locus model
muvar(n.loci=1,y1=c(0,1,1),p1=0.5)

# the default 2-locus model
muvar(n.loci=2,y12=c(1,1,1,1,0,0,0,0),p1=0.99,p2=0.9)

## End(Not run)
```

mvmeta

Multivariate meta-analysis based on generalized least squares

Description

This function accepts a data matrix of parameter estimates and their variance-covariance matrix from individual studies and obtain a generalized least squares (GLS) estimate and heterogeneity statistic.

For instance, this would be appropriate for combining linear correlation coefficients of single nucleotide polymorphisms (SNPs) for a given region.

Usage

mvmeta(b,V)

Arguments

- `b`: the parameter estimates
- `V`: the triangular variance-covariance matrix
Value

The returned value is a list containing:

- `d` the compact parameter estimates
- `Psi` the compact covariance-covariance matrix
- `X` the design matrix
- `beta` the pooled parameter estimates
- `cov.beta` the pooled variance-covariance matrix
- `X2` the Chi-squared statistic for heterogeneity
- `df` the degrees(s) of freedom
- `p` the p value

References


Author(s)

Jing Hua Zhao

See Also

`metareg`

Examples

```r
## Not run:
# example 11.3 from Hartung et al.
#
b <- matrix(c(
0.808, 1.308, 1.379, NA, NA,
NA, 1.266, 1.828, 1.962, NA,
NA, 1.835, NA, 2.568, NA,
NA, 1.272, NA, NA, 2.038,
1.171, 2.024, 2.423, 3.159, NA,
0.681, NA, NA, NA,NA),ncol=5, byrow=TRUE)

psi1 <- psi2 <- psi3 <- psi4 <- psi5 <- psi6 <- matrix(0,5,5)

psi1[1,1] <- 0.0985
psi1[1,2] <- 0.0611
psi1[1,3] <- 0.0623
psi1[2,2] <- 0.1142
psi1[2,3] <- 0.0761
psi1[3,3] <- 0.1215

psi2[2,2] <- 0.0713
psi2[2,3] <- 0.0539
psi2[2,4] <- 0.0561
```
Description

This is a study of the neprilysin gene and sporadic Alzheimer’s disease in Chinese. There are 257 cases and 242 controls, each with eight SNPs detecting through denaturing high-performance liquid chromatography (DHPLC).

Usage

data(nep499)

Format

A data frame
Source

pbsize

Description
This function implements Long et al. (1997) statistics for population-based association design. This is based on a contingency table test and accurate level of significance can be obtained by Fisher’s exact test.

Usage

pbsize(kp, gamma=4.5, p=0.15, alpha=5e-8, beta=0.2)

Arguments

kp population disease prevalence
 gamma genotype relative risk assuming multiplicative model
 p frequency of disease allele
 alpha type I error rate
 beta type II error rate

Value
The returned value is scaler containing the required sample size

References

Note
extracted from rm.c

Author(s)
Jing Hua Zhao
See Also

fbsize

Examples

kp <- c(0.01, 0.05, 0.10, 0.2)
models <- matrix(c(
  4.0, 0.01,
  4.0, 0.10,
  4.0, 0.50,
  4.0, 0.80,
  2.0, 0.01,
  2.0, 0.10,
  2.0, 0.50,
  2.0, 0.80,
  1.5, 0.01,
  1.5, 0.10,
  1.5, 0.50,
  1.5, 0.80), ncol=2, byrow=TRUE)
outfile <- "pbsize.txt"
cat("gamma", "p", "p1", "p5", "p10", "p20\n", sep="\t", file=outfile)
for(i in 1:dim(models)[1])
{
  g <- models[i,1]
p <- models[i,2]
  n <- vector()
  for(k in kp) n <- c(n, ceiling(pbsize(k, g, p)))
  cat(models[i,1:2], n, sep="\t", file=outfile, append=TRUE)
  cat("\n", file=outfile, append=TRUE)
}
table5 <- read.table(outfile, header=TRUE, sep="\t")
unlink(outfile)

# Alzheimer's disease
g <- 4.5
p <- 0.15
alpha <- 5e-8
beta <- 0.2
zalpha <- qnorm(1-alpha/2)  # 5.45
zbeta <- qnorm(1-beta)
q <- 1-p
pi <- 0.065                  # 0.07 and zbeta generate 163
k <- pi*(g*p+q)^2
s <- (1-pi*g^2)*p^2+(1-pi*g)*2*p+q+(1-pi)*q^2
# LGL formula
lambda <- pi*(g^2*p+q-(g*p+q)^2)/(1-pi*(g*p+q)^2)
# mine
lambda <- pi*p*q*(g-1)^2/(1-k)
n <- (zalpha+zbeta)^2/lambda
cat("\nPopulation-based result: Kp =", k, "Kq =", s, "n =", ceiling(n), "\n")
Description

This is a revised version of \texttt{pbsize} which is appropriate for a case-control design under a range of disease models. Essentially, for given sample size(s), a proportion of which (fc) being cases, the function calculates power estimate for a given type I error (alpha), genotype relative risk (gamma), frequency of the risk allele (p), the prevalence of disease in the population (kp) and optionally a disease model (model). A major difference would be the consideration of case/control ascertainment in \texttt{pbsize}.

Internally, the function obtains a baseline risk to make the disease model consistent with \texttt{Kp} as in \texttt{tscc} and should produce accurate power estimate. Note it provides power estimates for given sample size(s) only.

Usage

\begin{verbatim}
pbsize2(N,fc=0.5,alpha=0.05,gamma=4.5,p=0.15,kp=0.1,model="additive")
\end{verbatim}

Arguments

\begin{itemize}
\item \texttt{N} The sample size
\item \texttt{fc} The proportion of cases in the sample
\item \texttt{alpha} Type I error rate
\item \texttt{gamma} The genotype relative risk (GRR)
\item \texttt{p} Frequency of the risk allele
\item \texttt{kp} The prevalence of disease in the population
\item \texttt{model} Disease model, i.e., "multiplicative","additive","dominant","recessive","overdominant"
\end{itemize}

Value

The returned value is the power for the specified design.

Note

Why is the comparison with power.casectrl so bad?

Author(s)

Jing Hua Zhao

See Also

The design follows that of \texttt{pbsize}. 
Examples

## Not run:

```r
# single calc
m <- c("multiplicative","recessive","dominant","additive","overdominant")
for(i in 1:5) print(pbsize2(N=50, alpha=.1, gamma=1, p=0.1, kp=0.1, model=m[i]))

# for a range of sample sizes
pbsize2(p=0.1, N=(25,50,100,200,500), gamma=1, kp=.1, alpha=5e-2, model='r')

# create a power table
f <- function(p)
  pbsize2(p=p, N=seq(100,1000,by=100), gamma=1, kp=.1, alpha=5e-2, model='recessive')
m <- sapply(X=seq(0.1,0.9, by=0.1), f)
colnames(m) <- seq(0.1,0.9, by=0.1)
rownames(m) <- seq(100,1000,by=100)
print(round(m,2))

library(genetics)
m <- c("multiplicative","recessive","dominant","partialrecessive")
for(i in 1:4) print(power.casectrl(p=0.1, N=50, gamma=1, kp=.1, alpha=5e-2, minh=m[i]))
power.casectrl(p=0.1, N=(25,50,100,200,500), gamma=1, kp=.1, alpha=5e-2, minh='r')
f <- function(p)
  power.casectrl(p=p, N=seq(100,1000,by=100), gamma=1, kp=.1, alpha=5e-2, minh='recessive')
m <- sapply(X=seq(0.1,0.9, by=0.1), f)
colnames(m) <- seq(0.1,0.9, by=0.1)
rownames(m) <- seq(100,1000,by=100)
print(round(m,2))

## End(Not run)
```

---

**PD**

*A study of Parkinson's disease and APOE, LRRK2, SNCA makers*

---

**Description**

A study of Parkinson’s disease and controls with APOE, LRRK2 markers rs10506151, rs10784486, rs1365763, rs1388598, rs1491938, rs1491941 and SNCA markers m770, int4 and SNCA. The column abc indicates if a subject is familial Parkinson’s (+), sporadic (-), or controls (Control). Races involved are American Indians (AI), African American (B), and the rest are Caucasians. Diagnosis also included possible (POS), probable (PRO) and definite PDs. AON is the age at onset.

**Usage**

`data(PD)`
**Format**

A data frame

**Source**

Prof Abbas Parsian at NIH

**References**

Parsian et al. ASHG 2005, Toronto

---

**pedtodot**  
*Converting pedigree(s) to dot file(s)*

**Description**

This function converts GAS or LINKAGE formatted pedigree(s) into .dot file for each pedigree to be used by dot in graphviz, which is a flexible package for graphics freely available.

Note that a single PostScript (PDF) file can be obtained by dot, fdp, or neato.

```
dot -Tps <dot file> -o <ps file>
```

or

```
fdp -Tps <dot file> -o <ps file>
```

or

```
neato -Tps <dot file> -o <ps file>
```

See relevant documentations for other formats.

To preserve the original order of pedigree(s) in the data, you can examine the examples at the end of this document.

Under Cygwin/Linux/Unix, the PostScript file can be converted to Portable Document Format (PDF) default to Acrobat.

```
ps2pdf <ps file>
```

Use ps2pdf12, ps2pdf13, or ps2pdf14 for appropriate versions of Acrobat according to information given on the headline of <ps file>.

Under Linux, you can also visualize the .dot file directly via command,

```
dotty <dot file> 
```

**Usage**

```
pedtodot(pedfile,makeped=FALSE,sink=TRUE,page="B5",url="http://www.mrc-epid.cam.ac.uk",height=0.5,width=0.75,rotate=0,dir="none")
```
Arguments

- **pedfile**: a pedigree file in GAS or LINKAGE format, note if individual's ID is character then it is necessary to specify as.is=T in the read.table command
- **makeped**: a logical variable indicating if the pedigree file is post-makeped
- **sink**: a logical variable indicating if .dot file(s) are created
- **page**: a string indicating the page size, e.g., A4, A5, B5, Legal, Letter, Executive, "x,y", where x, y is the customized page size
- **url**: Unified Resource Locator (URL) associated with the diagram(s)
- **height**: the height of node(s)
- **width**: the width of node(s)
- **rotate**: if set to 90, the diagram is in landscape
- **dir**: direction of edges, i.e., "none", "forward", "back", "both". This will be useful if the diagram is viewed by Ineato

Details

We can extract the code below (or within pedtodot.Rd) to pedtodot and then use command:

```
sh pedtodot <pedigree file>
```

Value

For each pedigree, the function generates a .dot file to be used by dot. The collection of all pedigrees (*.dot) can also be put together.

Note

This is based on the gawk script program pedtodot by David Duffy with minor changes

Author(s)

David Duffy, Jing Hua Zhao

See Also

package sem in CRAN and Rgraphviz in BioConductor [http://www.bioconductor.org](http://www.bioconductor.org)

Examples

```r
## Not run:
# example as in R News and Bioinformatics (see also plot.pedigree in package kinship)
# it works from screen paste only
pl <- scan(nlines=16,what=list(0,0,0,0,0,0,"",""))
  1 2 3 2 2 7/7 7/10
  2 0 0 1 1  / /  / /
  3 0 0 2 2 7/9 3/10
  4 2 3 2 2 7/9 3/7
  5 2 3 2 1 7/7 7/10
```
p2 <- as.data.frame(p1)
names(p2) <- c("id","fid","mid","sex","aff","GABRB1","D4S1645")
p3 <- data.frame(pid=10081,p2)
attach(p3)
pedtodot(p3)
#
# Three examples of pedigree-drawing
# assuming pre-MakePed LINKAGE file in which IDs are characters
pre<-read.table("pheno.pre",as.is=TRUE)[,1:6]
pedtodot(pre)
dir()
# for post-MakePed LINKAGE file in which IDs are integers
ped <-read.table("pheno.ped")[,1:10]
pedtodot(ped,makeped=TRUE)
dir()
# for a single file with a list of pedigrees ordered data
sink("gaw14.dot")
pedtodot(ped,sink=FALSE)
sink()
file.show("gaw14.dot")
# more details
pedtodot(ped,sink=FALSE,page="B5",url="http://www.mrc-epid.cam.ac.uk/")

# An example from Richard Mott and in the demo
filespec <- system.file("tests/ped.1.3.pre")
pre <- read.table(filespec,as.is=TRUE)
pre
pedtodot(pre,dir="forward")

## End(Not run)

---

**pfc**  

**Probability of familial clustering of disease**

---

**Description**

To calculate exact probability of familial clustering of disease

```
6 2 3 1 1 7/7 7/10
7 2 3 2 1 7/7 7/10
8 0 0 1 1 --/ --/
9 8 4 1 1 7/9 3/10
10 0 0 2 1 --/ --/
11 2 10 2 1 7/7 7/7
12 2 10 2 2 6/7 7/7
13 0 0 1 1 --/ --/
14 13 11 1 1 7/8 7/8
15 0 0 1 1 --/ --/
16 15 12 2 1 6/6 7/7
```
Usage

pfc(famdata, enum)

Arguments

famdata  collective information of sib size, number of affected sibs and their frequencies
enum     a switch taking value 1 if all possible tables are to be enumerated

Value

The returned value is a list containing (tailp, sump, nenum are only available if enum=1):

p         the probability of familial clustering
stat      the deviances, chi-squares based on binomial and hypergeometric distributions,
          the degrees of freedom should take into account the number of marginals used
tailp     the exact statistical significance
sump      sum of the probabilities used for error checking
nenum     the total number of tables enumerated

References


Note

Adapted from family.for by Dani Zelterman, 25/7/03

Author(s)

Dani Zelterman, Jing Hua Zhao

See Also

kin.morgan

Examples

## Not run:
# IPF among 203 siblings of 100 COPD patients from Liang KY, SL Zeger,
# Qaquis B. Multivariate regression analyses for categorical data

# the degrees of freedom is 15
famtest<-c(
  1, 0, 36,
  1, 1, 12,
  2, 0, 15,
pfc.sim

Probability of familial clustering of disease

Description
To calculate probability of familial clustering of disease using Monte Carlo simulation

Usage
pfc.sim(famdata,n.sim=1000000,n.loop=1)

Arguments
famdata collective information of sib size, number of affected sibs and their frequencies
n.sim number of simulations in a single Monte Carlo run
n.loop total number of Monte Carlo runs

Value
The returned value is a list containing:
n.sim a copy of the number of simulations in a single Monte Carlo run
n.loop the total number of Monte Carlo runs
p the observed p value
tailpl accumulated probabilities at the lower tails
tailpu simulated p values

2, 1, 7,
2, 2, 1,
3, 0, 5,
3, 1, 7,
3, 2, 3,
3, 3, 2,
4, 0, 3,
4, 1, 3,
4, 2, 1,
5, 0, 1,
5, 2, 1,
5, 3, 1,
5, 4, 1,
6, 6, 1

test<-t(matrix(famtest,nrow=3))
famp<-pfc(test)

## End(Not run)
References


Note

Adapted from runi.for from Change Yu, 5/6/4

Author(s)

Chang Yu, Dani Zelterman

See Also

pfc

Examples

## Not run:
# Li FP, Fraumeni JF Jr, Mulvihill JJ, Blattner WA, Dreyfus MG, Tucker MA,
# Miller RW. A cancer family syndrome in twenty-four kindreds.

# family_size  #_of_affected frequency

famtest<-c(
  1, 0, 2,
  1, 1, 0,
  2, 0, 1,
  2, 1, 4,
  2, 2, 3,
  3, 0, 0,
  3, 1, 2,
  3, 2, 1,
  3, 3, 1,
  4, 0, 0,
  4, 1, 2,
  5, 0, 0,
  5, 1, 1,
  6, 0, 0,
  6, 1, 1,
  7, 0, 0,
  7, 1, 1,
  8, 0, 0,
  8, 1, 1,
  8, 2, 1,
  8, 3, 1,
  9, 3, 1)

test<-matrix(famtest,byrow=T,ncol=3)
famp<-pfc.sim(test)
Description

This function is a R port of the GENECOUNTING/PREPARE program which takes an array of genotypic data and collapses individuals with the same multilocus genotype. This function can also be used to prepare for the genotype table in testing Hardy-Weinberg equilibrium.

Usage

pgc(data, handle.miss=1, is.genotype=0, with.id=0)

Arguments

data | the multilocus genotype data for a set of individuals
handle.miss | a flag to indicate if missing data is kept, 0 = no, 1 = yes
is.genotype | a flag to indicate if the data is already in the form of genotype identifiers
with.id | a flag to indicate if the unique multilocus genotype identifier is generated

Value

The returned value is a list containing:

cdata | the collapsed genotype data
wt | the frequency weight
obscom | the observed number of combinations or genotypes
idsave | optional, available only if with.id = 1

References

Zhao JH, Sham PC (2003). Generic number system and haplotype analysis. Comp Prog Meth Biomed 70:1-9

Note

Built on pgc.c

Author(s)

Jing Hua Zhao
**plot.hap.score**

Plot haplotype frequencies versus haplotype score statistics

---

**Description**

Method function to plot a class of type hap.score

**Usage**

```r
## S3 method for class 'hap.score'
plot(x, ...)
```

**Arguments**

- `x` The object returned from hap.score (which has class hap.score).
- `...` Optional arguments

**Details**

This is a plot method function used to plot haplotype frequencies on the x-axis and haplotype-specific scores on the y-axis. Because hap.score is a class, the generic plot function can be used, which in turn calls this plot.hap.score function.

**Value**

Nothing is returned.

---

**See Also**

`genecounting.hwe.hardy`

**Examples**

```r
## Not run:

data(hla)
x <- hla[,3:8]

# do not handle missing data
y<-pgc(x,handle.miss=0,with.id=1)
hla.gc<-genecounting(y$data,y$wt,handle.miss=0)

# handle missing but with multilocus genotype identifier
pgc(x,handle.miss=1,with.id=1)

# handle missing data with no identifier
pgc(x,handle.miss=1,with.id=0)

## End(Not run)
```
print.hap.score

References

Schaid DJ, Rowland CM, Tines DE, Jacobson RM, Poland GA (2002) Score tests for association of traits with haplotypes when linkage phase is ambiguous. Amer J Hum Genet 70:425-34

See Also

hap.score

Examples

```r
## Not run:
save <- hap.score(y, geno, trait.type = "gaussian")

# Example illustrating generic plot function:
plot(save)

# Example illustrating specific method plot function:
plot.hap.score(save)

## End(Not run)
```

print.hap.score  
*Print a hap.score object*

Description

Method function to print a class of type hap.score

Usage

```r
## S3 method for class 'hap.score'
print(x, ...)
```

Arguments

- `x` The object returned from hap.score (which has class hap.score).
- `...` Optional arguments.

Details

This is a print method function used to print information from hap.score class, with haplotype-specific information given in a table. Because hap.score is a class, the generic print function can be used, which in turn calls this print.hap.score function.

Value

Nothing is returned.
References

Schaid DJ, Rowland CM, Tines DE, Jacobson RM, Poland GA (2002) Score tests for association of traits with haplotypes when linkage phase is ambiguous. Amer J Hum Genet 70:425-34

See Also

hap.score

Examples

```r
## Not run:
save <- hap.score(y, geno, trait.type = "gaussian")

# Example illustrating generic print function:
print(save)

# Example illustrating specific method print function:
print.hap.score(save)

## End(Not run)
```

qqfun

Quantile-comparison plots

Description

Plots empirical quantiles of a variable against theoretical quantiles of a comparison distribution.

Usage

```r
qqfun(x, distribution="norm", ylab=deparse(substitute(x)),
      xlab=paste(distribution, "quantiles"), main=NULL, las=par("las"),
      envelope=.95, labels=FALSE, col=palette()[4], lcol=palette()[2],
      xlim=NULL, ylim=NULL, lwd=1, pch=1, bg=palette()[4], cex=.4,
      line=c("quartiles", "robust", "none"), ...)```

Arguments

- `x` vector of numeric values.
- `distribution` root name of comparison distribution – e.g., `norm` for the normal distribution; `t` for the t-distribution.
- `ylab` label for vertical (empirical quantiles) axis.
- `xlab` label for horizontal (comparison quantiles) axis.
- `main` label for plot.
- `envelope` confidence level for point-wise confidence envelope, or `FALSE` for no envelope.
- `labels` vector of point labels for interactive point identification, or `FALSE` for no labels.
las if 0, ticks labels are drawn parallel to the axis; set to 1 for horizontal labels (see
par).
col color for points; the default is the fourth entry in the current color palette (see
palette and par).
lcol color for lines; the default is the second entry as above.
xlim the x limits (x1, x2) of the plot. Note that x1 > x2 is allowed and leads to a
reversed axis.
ylim the y limits of the plot
pch plotting character for points; default is Q (a circle, see par).
bg background color of points
cex factor for expanding the size of plotted symbols; the default is .4.
lwd line width; default is 1 (see par). Confidence envelopes are drawn at half this
line width.
line "quartiles" to pass a line through the quartile-pairs, or "robust" for a robust-
regression line; the latter uses the rlm function in the MASS package. Specifying
line = "none" suppresses the line.
... arguments such as df to be passed to the appropriate quantile function.

Details

Draws theoretical quantile-comparison plots for variables and for studentized residuals from a linear
model. A comparison line is drawn on the plot either through the quartiles of the two distributions,
or by robust regression.

Any distribution for which quantile and density functions exist in R (with prefixes q and d, respectively) may be used. Studentized residuals are plotted against the appropriate t-distribution.

This is adapted from qq.plot of package car with different values for points and lines, more op-
tions, more transparent code and examples in the current setting. Another similar but sophisticated
function is qqmath of package lattice.

Value

NULL. These functions are used only for their side effect (to make a graph).

Author(s)

John Fox, Jing Hua Zhao

References


See Also

qqnorm, qqunif, gcontrol2
qqunif

Examples

```r
## Not run:
p <- runif(100)
alpha <- 1/log(10)
qqfun(p,dist="unif")
qqfun(-log10(p),dist="exp",rate=alpha,pch=21)

#library(car)
#qq.plot(p,dist="unif")
#qq.plot(-log10(p),dist="exp",rate=alpha)

#library(lattice)
#qqplot(~ -log10(p), distribution = function(p) qexp(p,rate=alpha))

## End(Not run)
```

qqunif  

Q-Q plot for uniformly distributed random variable

Description

This function produces Q-Q plot for a random variable following uniform distribution with or without using log-scale. Note that the log-scale is by default for type "exp", which is a plot based on exponential order statistics. This appears to be more appropriate than the commonly used procedure whereby the expected value of uniform order statistics is directly log-transformed.

Usage

```r
qqunif(u,type="unif",logscale=TRUE,base=10,
col=palette()[4],lcol=palette()[2],ci=FALSE,alpha=0.05,...)
```

Arguments

- `u`: a vector of uniformly distributed random variables
- `type`: string option to specify distribution: "unif"=uniform, "exp"=exponential
- `logscale`: to use logscale
- `base`: the base of the log function
- `col`: color for points
- `lcol`: color for the diagonal line
- `ci`: logical option to show confidence interval
- `alpha`: 1-confidence level, e.g., 0.05
- `...`: other options as appropriate for the qqplot function
Value

The returned value is a list with components of a qqplot:

- **x**: expected value for uniform order statistics or its \( -\log(\text{base}) \) counterpart
- **y**: observed value or its \( -\log(\text{base}) \) counterpart

References


Author(s)

Jing Hua Zhao

See Also

qqfun

Examples

```r
## Not run:
# Q-Q Plot for 1000 U(0,1) r.v., marking those <= 1e-5
u_obs <- runif(1000)
r <- qqunif(u_obs,pch=21,bg="blue",bty="n")
u_exp <- r$y
hits <- u_exp >= 2.30103
points(r$x[hits],u_exp[hits],pch=21,bg="green")

## End(Not run)
```

---

**read.ms.output**  
A utility function to read ms output

Description

This function reads in the output of the program ms, a program to generate samples under a variety of neutral models.

The argument indicates either a file name or a vector of character strings, one string for each line of the output of ms. As with the second case, it is appropriate with `system intern=TRUE`), see example below.

Usage

```r
read.ms.output(msout,is.file=TRUE,xpose=TRUE,verbose=TRUE,
               outfile=NULL,outfileonly=FALSE)
```
read.ms.output

Arguments

msout an ms output
is.file a flag indicating ms output as a system file or an R object
xpose a flag to obtain the tranposed format as it is (when TRUE)
verbose when TRUE, display on screen every 1000 for large nsam
outfile to save the haplotypes in a tab-delimited ASCII file
outfileonly to reset gametes to NA when nsam/nreps is very large and is useful with outfile

Value

The returned value is a list storing the results.
call system call to ms
seed random number seed to ms
nsam number of copies of the locus in each sample
nreps the number of independent samples to generate
segsites a vector of the numbers of segregating sites
times vectors of time to most recent ancester (TMRCA) and total tree lengths
positions positions of polymorphic sites on a scale of (0,1)
gametes a list of haplotype arrays
probs the probability of the specified number of segregating sites given the genealogical history of the sample and the value to -t option

References

Hudson RR (2002) Generating samples under a Wright-Fisher neutral model. Bioinformatics 18:337-8,

Author(s)

D Davison, RR Hudson, JH Zhao

Examples

## Not run:

# Assuming ms is on the path

system("ms 5 4 -s 5 > ms.out")
msout1 <- read.ms.output("ms.out")

system("ms 50 4 -s 5 > ms.out")
msout2 <- read.ms.output("ms.out", outfile="out", outfileonly=TRUE)
s2k

Statistics for 2 by K table

Description

This function calculates one-to-others and maximum accumulated chi-squared statistics for a 2 by K contingency table.

Usage

s2k(y1,y2)

Arguments

y1 a vector containing the first row of a 2 by K contingency table
y2 a vector containing the second row of a 2 by K contingency table

Value

The returned value is a list containing:

x2a the one-to-other chisquare
x2b the maximum accumulated chisquare
col1 the column index for x2a
col2 the column index for x2b
p the corresponding p value

References


Note

The lengths of y1 and y2 should be the same

Author(s)

Chihiro Hirotsu, Jing Hua Zhao
Examples

```r
## Not run:
# an example from Mike Neale
# termed 'ugly' contingency table by Patrick Sullivan
y1 <- c(2, 15, 16, 35, 132, 30, 25, 7, 12, 24, 10, 10, 0)
y2 <- c(0, 6, 31, 49, 120, 27, 15, 8, 14, 25, 3, 9, 3)
result <- s2k(y1, y2)
## End(Not run)
```

---

### Description

**snp.PAR** gives PAR for a particular SNP.

**snp.ES** gives the effect size estimate based on the linear regression coefficient and standard error. For logistic regression, we can have similar idea for log(OR) and log(SE(OR)).

**snp.HWE** gives an exact Hardy-Weinberg Equilibrium (HWE) test, and -1 in the case of mis specification of genotype counts.

Eventually, this will be a set of functions specifically for single nucleotide polymorphisms (SNPs), which are biallelic markers. This is particularly relevant to genomewide association studies (GWAS) using GeneChips and in line with the classic generalised single-locus model. **snp.HWE** is from Abecasis’s website and yet to adapt for chromosome X.

Internally, **snp.PAR** calls for an internal function **PARn**, which calculates the population attributable risk (PAR) given a set of frequencies and associate relative risks (RR). Other 2x2 table statistics familiar to epidemiologists can be added when necessary.

### Usage

```r
snp.ES(beta, SE, N)
snp.HWE(g)
snp.PAR(RR, MAF, unit=2)
```

### Arguments

- **MAF**: Minor allele frequency
- **RR**: Relative risk
- **unit**: Unit to exponentiate for homozygote
- **beta**: Regression coefficient
- **SE**: Standard error for beta
- **N**: Sample size
- **g**: Observed genotype vector
**Power calculation for two-stage case-control design**

This function gives power estimates for two-stage case-control design for genetic association. The false positive rates are calculated as follows,

\[ P(|z_1| > C_1)P(|z_2| > C_2, \text{sign}(z_1) = \text{sign}(z_2)) \]

and

\[ P(|z_1| > C_1)P(|z_j| > C_j|z_1| > C_1) \]

for replication-based and joint analyses, respectively; where \(C_1, C_2,\) and \(C_j\) are thresholds at stages 1, 2 replication and joint analysis,

\[
z_1 = z(p_1, p_2, n_1, n_2, \pi_{\text{samples}}) \\
z_2 = z(p_1, p_2, n_1, n_2, 1 - \pi_{\text{samples}}) \\
z_j = \sqrt{\pi_{\text{samples}}} * z_1 + \sqrt{1 - \pi_{\text{samples}}} * z_2
\]

**Usage**

\[
\text{tscc}(\text{model}, \text{GRR}, p_1, n_1, n_2, M, \alpha_{\text{genome}}, \pi_{\text{samples}}, \pi_{\text{markers}}, K)
\]

**Arguments**

- **model**: any in c("multiplicative","additive","dominant","recessive")
- **GRR**: genotype relative risk
- **p1**: the estimated risk allele frequency in cases
- **n1**: total number of cases
- **n2**: total number of controls
- **M**: total number of markers
- **alpha.genome**: false positive rate at genome level
- **pi.samples**: sample% to be genotyped at stage 1
- **pi.markers**: markers% to be selected (also used as the false positive rate at stage 1)
- **K**: the population prevalence
Value

The returned value is a list containing a copy of the input plus output as follows,

- **model**: any in c("multiplicative","additive","dominant","recessive")
- **GRR**: genotype relative risk
- **p1**: the estimated risk allele frequency in cases
- **pprime**: expected risk allele frequency in cases
- **p**: expected risk allele frequency in controls
- **n1**: total number of cases
- **n2**: total number of controls
- **M**: total number of markers
- **alpha.genome**: false positive rate at genome level
- **pi.samples**: sample% to be genotyped at stage 1
- **pi.markers**: markers% to be selected (also used as the false positive rate at stage 1)
- **K**: the population prevalence
- **C**: thresholds for no stage, stage 1, stage 2, joint analysis
- **power**: power corresponding to C

References


Note

solve.skol is adapted from CaTS

Author(s)

Jing Hua Zhao

Examples

```R
K <- 0.1
p1 <- 0.4
n1 <- 1000
n2 <- 1000
M <- 300000
alpha.genome <- 0.05
GRR <- 1.4
p1 <- 0.4
pi.samples <- 0.2
pi.markers <- 0.1

options(echo=FALSE)
cat("sample%,marker%,GRR,(thresholds x 4)(power estimates x 4)\n")
```
whscore

for(GRR in c(1.3, 1.35, 1.40)) {
cat("\n")
for(pi.samples in c(1.0, 0.5, 0.4, 0.3, 0.2)) {
  if(pi.samples==1.0) s <- 1.0
  else s <- c(0.1, 0.05, 0.01)
  for(pi.markers in s) {
    x <- tscc("multiplicative", GRR, pi.samples, pi.markers, K)
    l <- c(pi.samples, pi.markers, GRR, x$C, x$power)
    l <- sprintf("%.2f %.2f %.2f %.2f %.2f %.2f %.2f %.2f %.2f %.2f %.2f",
                 x[1], x[2], x[3], x[4], x[5], x[6], x[7], x[8], x[9], x[10], x[11])
    cat(l, "\n")
  }
cat("\n")
}
}
options(echo=TRUE)

whscore

Whittemore-Halpern scores for allele-sharing

Description

Allele sharing score statistics

Usage

whscore(allele, type)

Arguments

allele a matrix of alleles of affected pedigree members
type 0 = pairs, 1 = all

Value

The returned value is the value of score statistic

References


whscore

Note
adapted from GENEHUNTER

Author(s)
Leonid Kruglyak, Jing Hua Zhao

Examples
```r
## Not run:
c <- matrix(c(1,1,2,2,2), ncol=2)
whscore(c, type=1)
whscore(c, type=2)
```
```r
## End(Not run)
```
Index

*Topic datagen
  b2r, 11
gif, 41
kin.morgan, 58
makeped, 65
mvmeta, 89
*Topic datasets
  aldh2, 8
  apoeapoc, 9
cf, 19
crohn, 24
fa, 28
fsnps, 33
hla, 51
l51, 60
lukas, 65
mao, 67
meyer, 77
mfblong, 78
nep499, 91
PD, 95
*Topic distribution
  qqfun, 105
  qgunif, 107
*Topic dplot
  pedtodor, 96
*Topic hplot
  asplot, 10
  ESplot, 27
  mhtplot, 79
  mhtplot2, 81
  plot.hap.score, 103
  qgunif, 107
*Topic htest
  ab, 6
  AE3, 7
  chow.test, 20
  comp.score, 22
  gcp, 37
  h2, 43
  hwe, 53
  hwe.cc, 54
  hwe.hardy, 56
  klem, 59
  MCMCgrm, 72
  metap, 73
  mtdt2, 86
*Topic misc
  ccsize, 16
  fbsize, 28
  masize, 67
  pbsize, 92
  pbsize2, 94
  tscC, 112
*Topic models
  AE3, 7
  BFDP, 13
  bt, 15
  FPRP, 30
  gc.em, 33
  gcontrol, 35
  gcontrol2, 36
  gcp, 37
  genecounting, 39
  hap, 45
  hap.em, 47
  hap.score, 48
  LD22, 61
  LDk1, 63
  metareg, 75
  mtdt, 85
  mtdt2, 86
  muvar, 88
  pfc, 98
  pfc.sim, 100
  s2k, 110
*Topic package
  gap-package, 3
INDEX

*Topic print
  print.hap.score, 104
*Topic regression
  hap.score, 48
  htr, 51
  qqfun, 105
*Topic univar
  qqfun, 105
  qqunif, 107
*Topic utilities
  mia, 83
  pgc, 102
  read.ms.output, 108
  SNP, 111
  whscore, 114
*Topic utilities
  muvar, 88

ab, 6, 71
AE3, 7
aldh2, 8
apoeapoc, 9
asplot, 10
b2r, 11
BFDP, 13, 32
bt, 15, 86
ccsize, 7, 16
cf, 19
chow.test, 20
comp.score, 22
crohn, 24
ESplot, 27
fa, 28
fbsize, 28, 93
FPRP, 14, 30
fsnps, 33
gap (gap-package), 3
gap-package, 3
gc.em, 33, 40, 61
gcontrol, 35
gcontrol2, 36, 106
gcp, 37
genecounting, 33, 35, 38, 39, 46, 60, 61, 103
genotype, 57
gif, 41, 58
h2, 43
hap, 45, 48, 84
hap.em, 47
hap.score, 48, 52, 104, 105
hla, 51
htr, 21, 51
hwe, 53, 55, 57
hwe.cc, 54
hwe.hardy, 53, 56, 103
HWE.test, 57
kin.morgan, 58, 99
klem, 59
151, 60
LD22, 12, 61, 64
LDkl, 35, 40, 48, 62, 63
lukas, 65
makeped, 65
mao, 67
masize, 67
MCMCgrm, 72
metap, 73
metareg, 74, 75, 90
meyer, 77
mfblong, 78
mhtplot, 79, 81
mhtplot2, 81
mia, 83
mtdt, 16, 85, 87
mtdt2, 86
muvar, 88
mvmeta, 12, 89
nep499, 91
palette, 106
par, 106
pbsize, 17, 29, 92, 94
pbsize2, 94
PD, 95
pedtodot, 96
pfc, 42, 98, 101
pfc.sim, 100
pgc, 102
plot.hap.score, 103
pm (gcp), 37
pmplus (gcp), 37
print.hap.score, 104

qqfun, 105, 108
qqnorm, 106
qqunif, 80, 106, 107

read.ms.output, 108

s2k, 110
SNP, 111
snp.ES (SNP), 111
snp.HWE (SNP), 111
snp.PAR (SNP), 111

tscc, 94, 112

whscore, 114