Package 'DNAtools'

February 19, 2015

Type Package

Title Tools for analysing forensic genetic DNA data

Version 0.1-21

Date 2014-09-03

Author Torben Tvedebrink <tvede@math.aau.dk> and James Curran <j.curran@auckland.ac.nz>. Mikkel Meyer Andersen <mikl@math.aau.dk> assisted on the multicore/CPU implementation.

Maintainer Torben Tvedebrink <tvede@math.aau.dk>

Description Computationally efficient tools for comparing all pairs of profiles in a DNA database. The expectation and covariance of the summary statistic is implemented for fast computing. Routines for estimating proportions of close related individuals are available. The use of wildcards (also called F-designation) is implemented. Dedicated functions ease plotting the results.

Depends R (>= 2.10.0), Rsolnp, multicool

Imports Rcpp (>= 0.11.2)

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-09-03 18:07:11

R topics documented:

`DNAtools-package` *Tools for analysing forensic genetic DNA databases*

Description

Computational efficient tools for comparing all pairs of profiles in a DNA database. The expectation and covariance of the summary statistic is implemented for fast computing. Routines for estimating proportions of close related individuals are available. The use of wildcards (also called F-designation) is implemented. Dedicated functions ease plotting the results.

Details

Package:	DNAtools
Type:	Package
Version:	0.1
Date:	2014-08-25
License:	GPL (>= 2)

dbCompare: Compares make all n(n-1)/2 pairwise comparisons between profiles of a database with n DNA profiles. dbExpect: Computes the expected number of matching and partial matching loci for a given number of profiles in a database. dbVariance: Calculates the associated covariance matrix.

Author(s)

Torben Tvedebrink <tvede@math.aau.dk> and James Curran <j.curran@auckland.ac.nz> wrote the package. Mikkel Meyer Andersen <mikl@math.aau.dk> assisted on the multicore/CPU implementation.

Maintainer: Torben Tvedebrink <tvede@math.aau.dk>

References

Tvedebrink T, JM Curran, PS Eriksen, HS Mogensen and N Morling (2012). Analysis of matches and partial-matches in a Danish STR data set. Forensic Science International: Genetics, 6(3): 387-392.

Read the vignette: `vigette("DNAtools")`

Examples

```
  ## Not run:
  data(dbExample)
  dbCompare(dbExample,hit=5,trace=TRUE)

## End(Not run)
```

---

dbCollapse — *Collapse m/p output to vector*

Description

Collapse a m/p-matrix from dbCompare/dbExpect to a vector.

Usage

```
dbCollapse(x)
```

Arguments

`x`	Either a object of class "dbcompare" (result from dbCompare) or "matrix".

Details

Collapse a m/p-matrix from dbCompare/dbExpect to a vector with entry i being the sum of all entries from m/p-matrix satisfying 2*m+p=i.

Value

A vector of length 2*max(m)+1 with entries begin the sum of entries i in m/p-matrix satisfying i=2*m+p.

Author(s)

Torben Tvedebrink

Examples

```
## Not run:
data(dbExample)
res <- dbCompare(dbExample, hit=5, trace=TRUE)
dbCollapse(res) ## same as dbCompare(dbExample, hit=5, trace=TRUE, collapse=TRUE)

## End(Not run)
```

---

dbCompare *Compare DNA profiles*

Description

Compare DNA profiles

Usage

```
dbCompare(x, profiles=NULL, hit=7, trace=TRUE, vector=FALSE, collapse=FALSE,
          wildcard=FALSE,wildcard.effect=FALSE, wildcard.impose=FALSE,
          Rallele=FALSE, threads=2)
```

Arguments

x	Database with DNA profiles. The database format is expected to be a data frame with each column containing an allelic number such that for each DNA marker there are two columns in the data frame. See `data(dbExample)` for an example of the format.
profiles	One or more profiles to be compared with all profiles in the database. Input is a vector, matrix or data frame of same length/width as a row in the database x. If profiles is non-null only one CPU will be used. In case threads>1 a warning will be given but computations performed using single core.
hit	The number of matching loci for further investigation
trace	Shows a progress bar
vector	Logical. Whether the result should be returned as vector or a matrix. Note if 'collapse' is TRUE vector is ignored.
collapse	Logical (default FALSE). If TRUE the (m,p)-matrix will be collapased into a (2*m+p)-vector containing the total number of matching alleles.
wildcard	Use the wildcard comparing.
wildcard.effect	Compare result of wildcard and no wildcard.
wildcard.impose	Force homozygouse profiles (aa) to have wildcard (aF).
Rallele	Implementation of 'Rare allele'designation matching.
threads	The number of threads to use for performing comparisons in parallel for increased computation time. Use 0 for using the same number as the computer has CPU cores. NOTE: Only available on Linux and MacOS operating systems.

Details

Computes the distance between DNA profiles in terms of matching and partially-matching STR loci.

Value

Returns a matrix with the number of pairs mathcing/partially-matching at (i,j)-loci.

Author(s)

James Curran and Torben Tvedebrink. The multicore/CPU implementation was provided by Mikkel Meyer Andersen.

Examples

```
  ## Not run:
  data(dbExample)
  dbCompare(dbExample,hit=5,trace=TRUE)

## End(Not run)
```

---

dbExample *Simulated database with 1,000 individuals*

Description

Database containing 1,000 simulated DNA profiles typed on ten autosomal markers.

Usage

```
dbExample
```

Format

A data frame with each row being a DNA profile and each column a part of a genetic marker. Note that homozygote profiles has the same allelic value in the two columns associated to the same marker.

dbExpect *Expected value of cell counts in DNA database comparison*

Description

Computes the expected number of cell counts when comparing DNA profiles in a DNA database. For every pair of DNA profiles in a database the number of matching and partial matching loci is recorded. A match is declared if the two DNA profiles coincide for both alleles in a locus and a partial-match is recorded if only one allele is shared between the profiles. With a total of L loci the number of matching loci is 0,...,L and partial number of matches is 0,...,L-m, where m is the number of matching loci.

Usage

```
dbExpect(probs,theta=0,k=c(0,0,1),n=1,r=0,R=0,round=FALSE,na=TRUE,vector=FALSE,
         collapse=FALSE,wildcard=FALSE,no.wildcard=NULL,
         rare.allele=FALSE,no.rare.allele=NULL)
```

Arguments

`probs`	List of vectors with allele probabilities for each locus
`theta`	The coancestery coefficient
`k`	The vector of identical-by-descent probabilities, k=(k2,k1,k0), where for full-siblings k=c(1,2,1)/4. The default is k=c(0,0,1) refering to unrelated individuals.
`n`	Number of DNA profiles in the database
`r`	The probability assigned to the rare alleles (see rare allele matching). If a vector must be of same length as `probs`.
`R`	The probability assigned to alleles shorter or longer than allelic ladder (see rare allele matching). If a vector must be of length 1 or 2, and if a list it must be same length as `probs`.
`round`	Whether or not the results should be rounded or not
`na`	Whether or not the off-elements should be returned as 0 or NA
`vector`	Whether or not the result should be returned as a matrix or vector. Note if 'collapse' is TRUE vector is ignored.
`collapse`	Logical (default FALSE). If TRUE the (m,p)-matrix will be collapased into a (2*m+p)-vector containing the total number of matching alleles.
`wildcard`	Should wildcards be used?
`no.wildcard`	Should 'w' wildcards be used?
`rare.allele`	Should rare allele matching be used?
`no.rare.allele`	Should 'r' rare allele loci be used?

Details

Computes the expected cell counts using a recursion formula. See Tvedebrink et al (2011) for details.

Value

Returns a matrix (or vector, see above) of expected cell counts.

Author(s)

James Curran and Torben Tvedebrink

References

T Tvedebrink, PS Eriksen, J Curran, HS Mogensen, N Morling. 'Analysis of matches and partial-matches in Danish DNA reference profile database'. Forensic Science International: Genetics, 2011.

Examples

```
  ## Not run:
  ## Simulate some allele frequencies:
  freqs <-  replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
              simplify=FALSE)
  ## Compute the expected number for a DB with 10000 profiles:
  dbExpect(freqs,theta=0,n=10000)

## End(Not run)
```

---

dbSimulate *Simulate a DNA database*

Description

Simulates a DNA database given a set of allele probabilities and theta value. It is possible to have close relatives in the database simulated in pairs, such that within each pair the profiles are higher correlated due to close familial relationship, but between pairs of profiles the correlation is only modelled by theta.

Usage

```
dbSimulate(probs,theta=0,n=1000,relatives=NULL)
```

Arguments

probs	List of allele probabilities, where each element in the list is a vector of allele probabilities.
theta	The coancestry coefficient
n	The number of profiles in the database
relatives	A vector of length 4. Determining the number of PAIRS of profiles in the database: (FULL-SIBLINGS, FIRST-COUSINS, PARENT-CHILD, AVUNCULAR). They should obey that 2*sum(relatives)<=n.

Details

Simulates a DNA database with a given number of DNA profiles (and possibly relatives) with a correlation between profiles governed by theta.

Value

A data frame where each row represents a DNA profile. The first column is a profile identifier (id) and the next 2*L columns contains the simulated genotype for each of the L loci. L is determined by the length of the list 'probs' with allele probabilities

Author(s)

James Curran and Torben Tvedebrink

Examples

```
## Not run:
## Simulate some allele frequencies:

freq <-  replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
           simplify=FALSE)
## Simulate a single database with 5000 DNA profiles:
simdb <- dbSimulate(freq,theta=0,n=5000)
## Simulate a number of databases, say N=50. For each database compute
## the summary statistic using dbCompare:
N <- 50
Msummary <- matrix(0,N,(length(freq)+1)*(length(freq)+2)/2)
for(i in 1:N)
  Msummary[i,] <- dbCompare(dbSimulate(freq,theta=0,n=1000),
                    vector=TRUE,trace=FALSE)$m
## Give the columns  representative names:
dimnames(Msummary)[[2]] <- DNAtools:::dbCats(length(freq),vector=TRUE)
## Plot the simulations using a boxplot
boxplot(log10(Msummary))
## There might come some warnings due to taking log10 to zero-values (no counts)
## Add the expected number to the plot:
points(1:ncol(Msummary),log10(dbExpect(freq,theta=0,n=1000,vector=TRUE)),
       col=2,pch=16)

## End(Not run)
```

dbVariance — *Covariance matrix of cell counts in DNA database comparison*

Description

Computes the covariance matrix for the cell counts when comparing DNA profiles in a DNA database. For every pair of DNA profiles in a database the number of matching and partial matching loci is recorded. A match is declared if the two DNA profiles coincide for both alleles in a locus and a partial-match is recorded if only one allele is shared between the profiles. With a total of L loci the number of matching loci is 0,...,L and partial number of matches is 0,...,L-m, where m is the number of matching loci. The expression is given by:

$$latex$$

Usage

```
dbVariance(probs,theta=0,n=1,collapse=FALSE)
```

Arguments

`probs`	List of vectors with allele probabilities for each locus
`theta`	The coancestery coefficient. If a vector of different theta values are supplied a list of covariance matrices is returned. Note it is faster to give a vector of theta values as argument than calculating each matrix at the time.
`n`	Number of DNA profiles in the database. If n=1 is supplied a list of the components for computing the variance is returned. That is, the variance and two covariances on the right hand side of the equation above.
`collapse`	Logical, default FALSE. If TRUE the covariance matrix is collapsed such that it relates to (2*m+p)-vectors of total number of matching alleles rather than (m,p)-matrix.

Details

Computes the covariance matrix of the cell counts using a recursion formula. See Tvedebrink et al (2011) for details.

Value

Returns a covariance matrix for the cell counts.

Author(s)

James Curran and Torben Tvedebrink

References

T Tvedebrink, PS Eriksen, J Curran, HS Mogensen, N Morling. 'Analysis of matches and partial-matches in Danish DNA reference profile database'. Forensic Science International: Genetics, 2011.

Examples

```
## Not run:
## Simulate some allele frequencies:
freqs <-  replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)}, simplify=FALSE)
## List of elements needed to compute the covariance matrix.
## Useful option when the covariance needs to be computed for varying
## database sizes but for identical theta-value.
comps <- dbVariance(freqs,theta=0,n=1)
## Covariance for a DB with 1000 DNA profiles
cov1000 <- dbVariance(freqs,theta=0,n=1000)
## The result is the same as:
comps1000 <- choose(1000,2)*comps$V1 + 6*choose(1000,3)*comps$V2 + 6*choose(1000,4)*comps$V3

## End(Not run)
```

---

estimatePD — *Estimate the drop-out probability based on number of alleles*

Description

An inferior may to estimate the drop-out probability compared to using the peak heights from the electropherogram. However, to compare the performance with Gill et al. (2007) this implements a theoretical approach based on their line of arguments.

Usage

```
estimatePD(n0,m,pnoa=NULL,probs=NULL,theta=0,noa.tabs=NULL,locuswise=FALSE)
```

Arguments

`n0`	Vector of observed allele counts - same length as the number of loci.
`m`	The number of contributors
`pnoa`	The vector of P(N(m)=n) for n=1,...,2Lm, where L is the number of loci and m is the number of contributors.
`probs`	List of vectors with allele probabilities for each locus
`theta`	The coancestery coefficient
`noa.tabs`	If `noa.tabs` has been computed by the `noaTabs`-function for `m`, then this will speed up the computations.
`locuswise`	Logical. Indicating whether computations should be done locuswise.

Details

Computes the P(D) that maximises equation (10) in Tvedebrink (2013).

Value

Returns the MLE of P(D) based on equation (10) in Tvedebrink (2013)

Author(s)

Torben Tvedebrink

References

Gill, P., A. Kirkham, and J. Curran (2007). LoComatioN: A software tool for the analysis of low copy number DNA profiles. Forensic Science International 166(2-3): 128 - 138.

T Tvedebrink (2013). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine: (under review).

Examples

```
## Simulate some allele frequencies:
freqs <-  structure(replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE),.Names=paste("locus",1:10,sep="."))
## Assume 15 alleles are observed in a 2-person DNA mixture with 10 loci:
estimatePD(n0=15,m=2,probs=freqs)
```

`freqEst` *Simple allele frequency estimation*

Description

Estimates allele frequencies from a database with DNA profiles

Usage

```
freqEst(x)
```

Arguments

`x`	A database of the form ["id","locus1 allele1","locus1 allele2",...,"locusN allele 1","locusN allele2"].

Details

Computes the allele frequencies for a given database.

Value

Returns a list of probability vectors - one vector for each locus.

Author(s)

James Curran and Torben Tvedebrink

Examples

```
data(dbExample)
freqEst(dbExample)
```

---

noaTabs — *Computes the alpha_m variants and their weights.*

Description

Computes the α_m-vectors and their weights, $c(\alpha_m)$, in order to compute the exact distribution of the number of alleles in a m-person DNA mixture.

Usage

```
noaTabs(alpha=NULL,m=2,weight=1)
```

Arguments

`alpha`	If `noaTabs` has been evaluated for `n<m`, then providing this object to `noaTabs` reduces the number of recursions needed to evaluate for `m`.
`m`	The number of contributors
`weight`	See the paper for details. Usually, this should be set to 1

Details

Computes the α_m-vectors and their weights, $c(\alpha_m)$, by recursion over the number of contributors m. That is, α_m is obtained based on α_{m-1} – see paper for details.

Value

Returns a named vector of counts, where the names denotes the α_m-vectors and the countes is the associated weights, $c(\alpha_m)$.

Author(s)

Torben Tvedebrink

References

T Tvedebrink (2013). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine: (under review).

Examples

```
## Simulate some allele frequencies:
freqs <-  structure(replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE),.Names=paste("locus",1:10,sep="."))
## Compute alpha_2
noa.tab2 <- noaTabs(m=2)
## Use alpha_2 to compute alpha_3
noa.tab3 <- noaTabs(alpha=noa.tab2, m=3)

## Compute P(N(m=3)=n), n=1,...,2*L*m, where L=10 here
pNoA(freqs,m=3,theta=0,noa.tabs=noa.tab3)
## Same, but locuswise results
pNoA(freqs,m=3,theta=0,noa.tabs=noa.tab3,locuswise=TRUE)
```

---

optim.relatedness — *Estimate theta and the fraction of comparisons between close relatives*

Description

Estimates the fraction of comparisons between pairs of close relatives while fitting the theta parameter minimising the object function. The function makes use of the R-package 'Rsolnp' which is an implementation of an solver for non-linear minimisation problems with parameter constraints.

Usage

```
optim.relatedness(obs,theta0=0,theta1=0.03,theta.tol=10^(-7),
                    theta.step=NULL,max.bisect=15,probs,var.list=NULL,
                    init.alpha=10^c(-4,-6,-8,-10),init.keep=FALSE,
                    objFunction=c("T2","T1","C3","C2","C1"),collapse=FALSE,
                    trace=FALSE,solnp.ctrl=list(tol=10^(-9),rho=10,
                    delta=min(init.alpha)*0.01,trace=FALSE))
```

Arguments

`obs`	The matrix or vector of observed matches/partial-matches as returned by the dbCompare()-function
`theta0`	The left value of the interval in which a bisection-like search is performed for theta
`theta1`	Right value of interval (see theta0)
`theta.tol`	A stopping criterion for the search. If the search narrows within theta.tol the function terminates

theta.step	Default is NULL. If not a grid search will be performed on seq(from = theta0, to = theta1, by = theta.step)
max.bisect	The maximum number of bisectional iterations perform prior to termination
probs	List of vectors with allele probabilities for each locus
var.list	A named list of components for computing variances, see dbVariance. The names of the elements are the associated theta-values, and each component is a list of (V1,V2,V3) - see dbVariance with n=1
init.alpha	Initial values for alpha, where the order is (First-cousins, Avuncular, Parent-child, Full-siblings). The value for Unrelated is computed as 1-sum(init.alpha)
init.keep	Whether the initial values should be used in successive steps for the current optimum should be used.
objFunction	Which of the five different object functions should be used to compare observed and expected
collapse	Not yet implemented
trace	Should iteration steps and other process indicators be printed
solnp.ctrl	See solnp for details

Details

Computes the proportion of comparisons between close relatives in a database matching exercise for each theta value under investigation.

Value

Returns a list of three components: value, solution and var.list. The first element, value, is a dataframe with the value of the objection function for each of the theta values investigated. Solution is the estimated alpha-vector where the objection function was minimised. Finally, var.list is a names list of components for computing variances. May be reused in later computations for increased speed in some iterations.

Author(s)

James Curran and Torben Tvedebrink

References

T Tvedebrink, PS Eriksen, J Curran, HS Mogensen, N Morling. 'Analysis of matches and partial-matches in Danish DNA reference profile database'. Forensic Science International: Genetics, 2011.

Examples

```
## Not run:
## Simulate some allele frequencies:
freqs <-  replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE)
## Load the sample database:
```

```
  data(dbExample)
  obs <- dbCompare(dbExample,trace=FALSE)$m
  C3 <- optim.relatedness(obs,theta0=0.0,theta1=0.03,probs=freqs,
          objFunction="C3",max.bisect=30,trace=TRUE)

## End(Not run)
```

---

pContrib — *Compute the posterior probabilities for P(m|n0) for a given prior P(m) and observed vector n0 of locus counts*

Description

where m ranges from 1 to $m_{\max}$ and n_0 is the observed locus counts.

Usage

```
pContrib(n0, probs=NULL, m.prior=rep(1/m.max,m.max), m.max=8, pnoa=NULL, theta=0)
```

Arguments

`n0`	Vector of observed allele counts - same length as the number of loci.
`probs`	List of vectors with allele probabilities for each locus
`m.prior`	A vector with prior probabilities (summing to 1), where the length of `m.prior` determines the plausible range of m
`m.max`	Derived from the length of `m.prior`, and if `m.prior=NULL` a uniform prior is speficied by `m.max`: `m.prior = rep(1/m.max,m.max)`.
`pnoa`	A named list of locus specific probabilities. Output from `pNoA` with `locuswise=TRUE`.
`theta`	The coancestery coefficient

Details

Computes a vector P(m|n0) evaluated over the plausible range 1,...,m.max.

Value

Returns a vector P(m|n0) for m=1,...,m.max

Author(s)

Torben Tvedebrink

References

T Tvedebrink (2013). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine: (under review).

Examples

```
## Simulate some allele frequencies:
freqs <-  structure(replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE),.Names=paste("locus",1:10,sep="."))
m <- 2
n0 <- unlist(lapply(freqs, function(x){
    length(unique(sample(length(x),size=2*m,replace=TRUE,prob=x)))}
    ))
## Compute P(m|n0) for m=1,...,4 and the sampled n0
pContrib(n0=n0,probs=freqs,m.max=4)
```

`pContrib.locus` *Compute the posterior probabilities for P(m|n0) for a given prior P(m)*

Description

Compute a matrix of posterior probabilties $P(m|n_0)$ where m ranges from 1 to $m_{\max}$, and n_0 is $0,\ldots,2m_{\max}$. This is done by evaluating $P(m|n_0) = P(n_0|m)P(m)/P(n)$, where $P(n_0|m)$ is evaluated by `pNoA`.

Usage

```
pContrib.locus(prob=NULL, m.prior=NULL, m.max=8, pnoa.locus=NULL, theta=0)
```

Arguments

`prob`	Vectors with allele probabilities for the specific locus
`m.prior`	A vector with prior probabilities (summing to 1), where the length of `m.prior` determines the plausible range of m
`m.max`	Derived from the length of `m.prior`, and if `m.prior=NULL` a uniform prior is speficied by `m.max`: `m.prior = rep(1/m.max,m.max)`.
`pnoa.locus`	A named vector of locus specific probabilities $P(N(m) = n), n = 1,\ldots,2m$.
`theta`	The coancestery coefficient

Details

Computes a matric of P(mln0) values for a specific locus.

Value

Returns a matrix [P(mln0)] for m=1,...,m.max and n0=1,...,2m.max.

Author(s)

Torben Tvedebrink

References

T Tvedebrink (2013). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine: (under review).

Examples

```
## Simulate some allele frequencies:
freqs <-  structure(replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE),.Names=paste("locus",1:10,sep="."))
## Compute P(m|n0) for m=1,...,5 and n0=1,...10 for the first locus:
pContrib.locus(prob=freqs[[1]],m.max=5)
```

---

plot.dbcompare *Plots the summary matrix*

Description

Plots the summary matrix with counts on y-axis and classification on x-axis.

Usage

```
## S3 method for class 'dbcompare'
plot(x,log="y",las=3,xlab="Match/Partial",ylab="Counts",...)
```

Arguments

x	Summary matrix returned from dbcompare
log	Specifies whether log(Counts) should be plotted (default)
las	Direction of the labels on x-axis. Default is 3 which gives perpendicular labels
xlab	Axis label
ylab	Axis label
...	Other plot options

Details

Plots the summary matrix

Value

A plot of the summary matrix. The counts are on log10 scale and the x-axis is labeled by appropriate matching/partially-matching levels.

Author(s)

James Curran and Torben Tvedebrink

See Also

dbCompare,print.dbcompare

Examples

```
  ## Not run:
  data(dbExample)
  M = dbCompare(dbExample,hit=5)
  plot(M)

## End(Not run)
```

`plot.dbOptim` *Plots the fitted object function for estimated familial relationships in the database and theta.*

Description

Plots the minimised object function for included values of theta

Usage

```
## S3 method for class 'dbOptim'
plot(x,type="l",...)
## S3 method for class 'dbOptim'
points(x,type="p",...)
## S3 method for class 'dbOptim'
lines(x,type="l",...)
```

Arguments

`x`	Object returned by optim.relatedness
`type`	The type of plot character ("l"=line, "p"=points, ...), see 'par' for more details
`...`	Other plot options

Details

Plots the object function

Value

A plot of the object function

Author(s)

James Curran and Torben Tvedebrink

See Also

optim.relatedness

Examples

```
## Not run:
## Simulate some allele frequencies:
freqs <-  replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE)
## Load the sample database:
data(dbExample)
obs <- dbCompare(dbExample,trace=FALSE)$m
C3 <- optim.relatedness(obs,theta0=0.0,theta1=0.03,probs=freqs,
        objFunction="C3",max.bisect=30,trace=TRUE)
plot(C3)

## End(Not run)
```

---

pNoA	*The exact distribution of the number of alleles in a m-person DNA mixture*

Description

Computes the exact distribution of the number of alleles in a m-person DNA mixture typed with STR loci. For a m-person DNA mixture it is possible to observe 1,...,2*m*L alleles, where L is the total number of typed STR loci. The method allows incorporation of the subpopulation correction, the so-called θ-correction, to adjust for shared ancestry. If needed, the locus-specific probabilities can be obtained using the `locuswise` argument.

Usage

```
pNoA(probs, m=2, theta = 0, noa.tabs=NULL, locuswise=FALSE)
```

Arguments

`probs`	List of vectors with allele probabilities for each locus
`m`	The number of contributors
`theta`	The coancestery coefficient
`noa.tabs`	If `noa.tabs` has been computed by the `noaTabs`-function for `m`, then this will speed up the computations.
`locuswise`	Logical. If TRUE the locuswise probabilities will be returned. Otherwise, the probability over all loci is returned.

Details

Computes the exact distribution of the number of alleles for a m-person DNA mixture.

Value

Returns a vector of probabilities, or a list of locuswise probability vectors.

Author(s)

Torben Tvedebrink

References

T Tvedebrink (2013). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine: (under review).

Examples

```
## Simulate some allele frequencies:
freqs <-  structure(replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
            simplify=FALSE),.Names=paste("locus",1:10,sep="."))
## Compute alpha_2
noa.tab3 <- noaTabs(m=2)

## Compute P(N(m=3)=n), n=1,...,2*L*m, where L=10 here
pNoA(freqs,m=2,theta=0,noa.tabs=noa.tab3)
## Same, but locuswise results
pNoA(freqs,m=2,theta=0,noa.tabs=noa.tab3,locuswise=TRUE)
```

---

print.dbcompare *Prints the summary matrix*

Description

Prints the summary matrix and possible "big hits".

Usage

```
## S3 method for class 'dbcompare'
print(x,...)
```

Arguments

`x`	Summary matrix returned from dbcompare
`...`	...

Details

Prints the summary matrix

Value

Prints the summary matrix and data frame with "big hits"

Author(s)

James Curran and Torben Tvedebrink

See Also

dbCompare,plot.dbcompare

Examples

```
## Not run:
data(dbExample)
M = dbCompare(dbExample,hit=5)
M

## End(Not run)
```

`print.dbOptim` *Prints the results from optim.relatedness()*

Description

Prints the evaluated functions for the object function, best estimate of alpha and possibly list of variances.

Usage

```
## S3 method for class 'dbOptim'
print(x,var.list=FALSE,...)
```

Arguments

`x`	Object returned by optim.relatedness()
`var.list`	Logical. Whether the (long) list of variance components should be printed to the screen.
`...`	...

Details

Prints the summary details of the fit

Value

A dataframe with [theta,value] and a vector of fitted alpha parameters

Author(s)

James Curran and Torben Tvedebrink

See Also

optim.relatedness

Examples

```
  ## Not run:
  ## Simulate some allele frequencies:
  freqs <-  replicate(10, { g = rgamma(n=10,scale=4,shape=3); g/sum(g)},
              simplify=FALSE)
  ## Load the sample database:
  data(dbExample)
  obs <- dbCompare(dbExample,trace=FALSE)$m
  C3 <- optim.relatedness(obs,theta0=0.0,theta1=0.03,probs=freqs,
          objFunction="C3",max.bisect=30,trace=TRUE)
  print(C3)

## End(Not run)
```

---

rcpp_hello_world *Simple function using Rcpp*

Description

Simple function using Rcpp

Usage

```
rcpp_hello_world()
```

Examples

```
## Not run:
rcpp_hello_world()

## End(Not run)
```

Index