Package ‘KinMixLite’

April 9, 2023

Type Package

Title Inference About Relationships from DNA Mixtures

Version 2.1.0

Date 2023-04-08

Maintainer Peter Green <`P.J.Green@bristol.ac.uk`>

Description Methods for inference about relationships between contributors to a DNA mixture and other individuals of known genotype: a basic example would be testing whether a contributor to a mixture is the father of a child of known genotype. This provides most of the functionality of the 'KinMix' package, but with some loss of efficiency and restriction on problem size, as the latter uses 'RHugin' as the Bayes net engine, while this package uses 'gRain'. The package implements the methods introduced in Green, P. J. and Mortera, J. (2017) <doi:10.1016/j.fsigen.2017.02.001> and Green, P. J. and Mortera, J. (2021) <doi:10.1111/rssc.12498>.

License GPL (>= 2)

Depends DNAmixturesLite, gRaven

Imports statnet.common, gRbase, Rsolnp, numDeriv, Matrix, ribd, pedtools, methods

URL `https://petergreenweb.wordpress.com/kinmix/`

NeedsCompilation no

Author Peter Green [aut, cre]

Repository CRAN

Date/Publication 2023-04-09 12:50:02 UTC

R topics documented:

`KinMixLite-package` *Inference About Relationships from DNA Mixtures*

Description

Methods for inference about relationships between contributors to a DNA mixture and other individuals of known genotype: a basic example would be testing whether a contributor to a mixture is the father of a child of known genotype. This provides most of the functionality of the 'KinMix' package, but with some loss of efficiency and restriction on problem size, as the latter uses 'RHugin' as the Bayes net engine, while this package uses 'gRain'. The package implements the methods introduced in Green, P. J. and Mortera, J. (2017) <doi:10.1016/j.fsigen.2017.02.001> and Green, P. J. and Mortera, J. (2021) <doi:10.1111/rssc.12498>.

Details

This package is a toolkit for inference about mixtures and familial relationships, either between contributors or between a contributor and other typed individuals. It extends the functionality proposed in Green and Mortera (2017) by allowing more general relationships, specified in general by an IBD pattern distribution - the generalisation to more than two individuals of the coefficients of identity of Jacquard (1974). Details are in the paper by Green and Mortera (2021). `KinMixLite` extends the capability of the **DNAmixturesLite** package, and intimately relies on that package; as with that package, instead of the **RHugin** package, it uses **gRaven** and **gRain** for Bayes Net calculations. This version implements the ALN, MBN and WLR as well as RPT methods; see Green and Mortera (2017).

Formats

See `formats` for formats of the various data objects used in this package.

Author(s)

Maintainer: Peter Green <P.J.Green@bristol.ac.uk>

References

Green, P. J. and Mortera, J. (2017). Paternity testing and other inference about relationships from DNA mixtures. *Forensic Science International: Genetics*. <doi:10.1016/j.fsigen.2017.02.001>.

Green, P. J. and Mortera, J. (2021). Inference about complex relationships using peak height data from DNA mixtures. *Applied Statistics*. <doi:10.1111/rssc.12498>.

Jacquard, A. (1974) *The genetic structure of populations*. Springer-Verlag.

See Also

`DNAmixturesLite`

Examples

```
require(ribd)
data(test2data)
data(NGMDyes)

C<-50

## Fit 2-person mixture - baseline model

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

## Fit 2-person mixture model in which contributor 1 is parent of a typed individual Cgt

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db,triangulate=FALSE,compile=FALSE)
rpt.IBD(mixD,'parent',list(c=Cgt),targets=c('f','c'),contrib='f')
log10LR<-(logL(mixD)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')

## Fit 2-person mixture, where contributors are siblings

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.6,U2=0.3,U3=0.1)))
baseline<-logL(mixD)(pars)
```

```
mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db,triangulate=FALSE,compile=FALSE)
rpt.IBD(mixD,'sibs',targets=c('b1','b2'),contribs=c('b1','b2'))
log10LR<-(protected(logL(mixD)(pars))-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

add.child.meiosis.nodes

Replace CPTs for mixture contributor a Father, given Child genotype, by MBN method

Description

loop over markers, and alleles within markers to create nodes for child allele count nodes, for paternity model with only Child genotyped then compile all domains. Implements method MBN.

Usage

```
add.child.meiosis.nodes(mixture,aca,ind=1)
```

Arguments

`mixture`	A compiled DNAmixture object
`aca`	Child's genotype profile as an allele count array
`ind`	Index of contributor regarded as Parent (or Child): which 'unknown' contributor are we modelling by amending his/her CPTs?

Details

To calculate the likelihood of this model, conditional on the child's genotype, a call to this function should be followed by (a) setting the finding of the child's genotype by defining `extra.findings`, (b) evaluating the loglikelihood using `logLX`, and (c) correcting the result by subtracting the log probability of the child's genotype, all as in the example below. Without (c), the value returned is the likelihood for the peak heights *and* the child's genotype.

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)

# set threshold C
C<-0.001

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

mixMBN<-DNAmixture(list(epg),k=2,C=list(C),database=db,triangulate=FALSE,compile=FALSE)
cgtcaca<-gt2aca(mixMBN,Cgt)
add.child.meiosis.nodes(mixMBN,cgtcaca,1)
log10LR<-(logLX(mixMBN,
expr.make.findings(list(
list('Male',ind=1),
list('Caca',aca='cgtcaca')
))
)(pars)-attr(cgtcaca,'logGt')-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

---

add.motherchild.likd.node

Replace CPTs for mixture contributor a Father, given Child and Mother genotypes, by ALN method

---

Description

loop over markers, and alleles within markers to create node Rlikd for relative likelihood for individual i, for paternity model with Mother and Child genotyped then compile all domains. Implements method ALN.

Usage

```
add.motherchild.likd.node(mixture,Cgt,Mgt,db,ind=1)
```

Arguments

`mixture`	A DNAmixture object
`Cgt`	Child's genotype profile as a data frame containing variables `marker`, `allele1` and `allele2`
`Mgt`	Mother's genotype profile as a data frame containing variables `marker`, `allele1` and `allele2`
`db`	Allele frequency database
`ind`	Index of contributor regarded as Father: which 'unknown' contributor are we modelling by amending his CPTs?

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)

# set threshold C
C<-0.001

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

mixD3<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,triangulate=FALSE,compile=FALSE)
cgtcaca<-gt2aca(mixD3,Cgt)
add.motherchild.likd.node(mixD3,Cgt,Mgt,db,1)
log10LR<-(logLX(mixD3,
expr.make.findings(list(
list('Male',ind=1),
list('Rlikd',aca='cgtcaca',cgt='Cgt',evid='Revid')
))
)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

---

add.relative.likd.node

Replace CPTs for mixture contributor a Father, given Child genotype, by ALN method

---

Description

loop over markers, and alleles within markers to create node Rlikd for relative likelihood for individual i, for paternity model with only Child genotyped then compile all domains. Implements method ALN.

Usage

```
add.relative.likd.node(mixture,aca,ind=1)
```

Arguments

`mixture`	A compiled DNAmixture object
`aca`	Child's genotype profile as an allele count array
`ind`	Index of contributor regarded as Parent (or Child): which 'unknown' contributor are we modelling by amending his/her CPTs?

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)

# set threshold C
C<-0.001

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

mixALN<-DNAmixture(list(epg),k=2,C=list(C),database=db,triangulate=FALSE,compile=FALSE)
cgtcaca<-gt2aca(mixALN,Cgt)
add.relative.likd.node(mixALN,cgtcaca,1)
log10LR<-(logLX(mixALN,
expr.make.findings(list(
list('Male',ind=1),
list('Rlikd',aca='cgtcaca',cgt='Cgt',evid='Revid')
))
)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

`as.gt` *Extract genotype profile for a single contributor from rGTs output*

Description

Extract genotype profile for a single contributor from rGTs output

Usage

```
as.gt(res,ind)
```

Arguments

`res`	Output from rGTs
`ind`	Integer, which individual's genotype profile should be extracted

Value

Data frame, genotype profile for selected individual, for format see `formats`.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

`checkpeaks` *Check and modify database to have positive frequencies for all observed peaks/alleles*

Description

Check whether database has positive frequencies for all peaks/alleles observed in epg and genotype profiles, and optionally modify db by addition of small positive frequencies so that it does, followed by renormalisation of frequencies for each allele to sum to 1.

Usage

```
checkpeaks(x,db,fix=0.003)
```

Arguments

`x`	data frame, the epg or genotype profile; see `formats`.
`db`	data frame, the db; see `formats`.
`fix`	numeric: if positive, increment to db frequency for each identified discrepant peak

Value

(possibly modified) db

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)

db<-checkpeaks(epg,db)
db<-checkpeaks(Cgt,db)

Xgt<-data.frame(marker=c('D10','D12'),allele1=c(8,13),allele2=c(13,10))
db<-checkpeaks(Xgt,db)
db
```

`convertIBD` *Convert relationship information to IBD pattern distribution*

Description

Construct IBD pattern distribution from one of several alternative representations of multi-person condensed coefficients of identity

Usage

```
as.IBD(x='sibs', targets=NULL, ped=FALSE)
convertIBD(x='sibs', targets=NULL, ped=FALSE)
```

Arguments

`x`	A string, a vector of length 3 or 9, a list with components `pr` and `patt`, or a list with two components, a pedigree and a vector of target id's; see Details
`targets`	character vector of individual tags
`ped`	logical, should complete pedigree be added as an attribute to the output, if available?

Details

Possible formats for the input `x` are:

1. certain verbal mnemonics; currently one of the following (or an unambiguous partial match): c('sibs','parent-child','half-sibs', 'cousins','half-cousins','second-cousins', 'double-first-cousins', 'quadruple-half-first-cousins', '3cousins-cyclic','3cousins-star','trio')
2. a vector of 3 kappas
3. a vector of 9 Deltas
4. a list with matrix or vector valued component `patt`, with or without component `pr`
5. a list with 2 components, the first being a pedigree in the sense of the `pedtools` package, the second a vector of target id's
6. a 3-column character matrix of individual tags, each row denoting a child/mother/triple - an alternative compact representation of a pedigree

Value

IBD pattern distribution as a list with components `pr` and `patt`

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

See Also

`pedtools`, `formats`

Examples

```
data(test2data)

IBD<-convertIBD('parent-child')

IBD<-convertIBD(c(0.5,0.5,0.0))
```

---

delete.DQnodes *Delete D and Q dummy nodes from all Bayes nets in mixture*

Description

Delete D and Q dummy nodes and associated edges from all Bayes nets in mixture, to save space; these nodes would only be needed for specific follow-up analyses

Usage

```
delete.DQnodes(mixture,which="DQ")
```

Arguments

`mixture`	A compiled DNAmixture object
`which`	character string

Details

The function removes the D and/or Q nodes from the DNAmixture object, depending on whether `which` includes "D", "Q" or both

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,dyes=list(NGMDyes),
triangulate=FALSE,compile=FALSE)
delete.DQnodes(mixD)
replace.tables.for.UAF(mixD,40)
size(mixD)
```

`emperors` *IBD pattern distribution in the Iulius-Claudius pedigree*

Description

IBD pattern distribution in the Iulius-Claudius pedigree

Usage

```
data("emperors")
```

Format

IBD object

See Also

`formats`.

Examples

```
data(emperors)
```

`expr.make.findings` *Coding additional findings as expression*

Description

Returns an expression that will be evaluated in logL.UKX whenever the likelihood of the model is calculated using the current method, and encodes the additional findings needed to implement the method; the details of the model and the extra information needed are held in the list z

Usage

```
expr.make.findings(z)
```

Arguments

`z`	A list specifying the additional findings; for the format, see Details

Details

Each component of the list `z` is a list encoding a particular type of additional finding: the first component of this (sub-)list being a character string specifying the type of finding, and the remainder of its components being named parameters giving details of the finding. The types of finding and the valid parameters of each are as follows:

`Male` `ind`: index of relevant contributor: which ‘unknown’ contributor are we modelling by amending his CPTs?

`Female` `ind`: index of relevant contributor

`Rlikd` `aca`: allele count array, `cgt`: character string naming genotype profile data frame, `evid`: character string naming list with one component for each marker, whose value is the evidence

`Aca` `ind`: index of relevant contributor, `aca`: allele count array

`Caca` `ind`: index of relevant contributor, `aca`: allele count array

`Denom` no parameters

If `z` is NULL, then there are no additional findings.

Value

Expression encoding the additional findings.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

`formats` *Formats*

Description

Formats for data objects in KinMix and KinMixLite

Formats

An **allele frequency database** is a data frame containing variables `marker`, `allele` and `frequency` (character, numeric and numeric respectively).

A **mixture profile** is a data frame containing variables `marker`, `allele` and `height` (character, numeric and numeric respectively).

A **genotype profile** is a data frame containing variables `marker`, `allele1` and `allele2` (character, numeric and numeric respectively).

Examples of these 3 data formats are objects `db`, `epg` and `Cgt`, respectively, in `test2data`.

A **allele count array** is an alternative format for a genotype as a named list of vectors, one for each marker. Each vector gives the number of each allele in the genotype, with the alleles listed in the order in which they appear in the `data` component of the relevant mixture object.

An **IBD pattern distribution** or **IBD object** is a list with components `pr` (a numerical vector) and `patt` (an integer matrix with `nrow(patt)==length(pr)` and an even number of columns). The elements of `pr` are the probabilities of the IBD patterns in the corresponding rows of `patt`. Adjacent pairs of columns encode the genotypes of different individuals; equal elements in any row determine equality of the alleles; different elements denote independent draws from the gene pool. If the component `pr` is missing, functions `rpt.IBD` and `rpt.typed.relatives` assume the probabilities are equal.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

`gt2aca` *Converts genotype profiles to allele count arrays*

Description

Returns list of vectors of allele counts corresponding to genotype profile in gt

Usage

```
gt2aca(mixture,gt,eps=0)
```

Arguments

`mixture`	A compiled DNAmixture object
`gt`	Genotype profile as a data frame containing variables `marker`, `allele1` and `allele2`
`eps`	If non-zero, the function creates the output allele count arrays in a different format, that mitigates subsequent propagation errors in some situations. Instead of a vector of allele counts, each element of the list is a matrix with 3 columns, corresponding to allele counts 0, 1 and 2, with entries 1 or `eps`.

Value

Returns list of vectors of allele counts. The log probability for the genotype is returned in its attribute '`logGt`'.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

# set threshold C
C<-0.001

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db,dyes=list(NGMDyes))
cgtcaca<-gt2aca(mixD,Cgt)

print(Cgt)
print(cgtcaca)
```

`intoMix` *Edit output from rGTs to omit individuals with NA amounts of DNA*

Description

Edit output from rGTs to omit individuals with NA amounts of DNA

Usage

```
intoMix(res)
```

Arguments

`res`	Output from `rGTs`

Value

The edited data structure

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

`logL.UKX` *Evaluates mixture log likelihood for unknown contributors with extra findings*

Description

Replacement for logL.UK in DNAmixtures that calls `extra.findings` immediately before propagating all findings and returning the normalising constant for the network.

Usage

```
logL.UKX(mixture, expr.extra.findings, initialize = FALSE)
```

Arguments

`mixture`	Compiled DNAmixture object.
`expr.extra.findings`	expression containing the extra findings
`initialize`	should all entered evidence be removed from the networks in `mixture`

Value

The log likelihood.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

See Also

See also `logL.UK`.

Examples

```
data(test2data)

# set threshold C
C<-0.001

pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.9,U2=0.1)))

mixMBN<-DNAmixture(list(epg),k=2,C=list(C),database=db,triangulate=FALSE,compile=FALSE)
cgtcaca<-gt2aca(mixMBN,Cgt)
add.child.meiosis.nodes(mixMBN,cgtcaca,1)
logL.UKX(mixMBN,
expr.make.findings(list(
list('Male',ind=1),
list('Caca',aca='cgtcaca')
)))(pars)
```

logLX — *Evaluates mixture log likelihood when extra findings present*

Description

Replacement for logL in DNAmixtures that calls calls LogL.UKX instead of logL.UK.

Usage

```
logLX(mixture, expr.extra.findings, presence.only = FALSE, initialize = FALSE)
```

Arguments

`mixture`	Compiled DNAmixture object.
`expr.extra.findings`	expression containing the extra findings
`presence.only`	Set to TRUE to ignore peak heights and evaluate the likelihood function considering peak presence and absence (heights above and below threshold) only. Defaults to FALSE
`initialize`	should all entered evidence be removed from the networks in `mixture`

Value

The log likelihood.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

See Also

See also `logL`.

Examples

```
data(test2data)

# set threshold C
C<-0.001

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

mixMBN<-DNAmixture(list(epg),k=2,C=list(C),database=db,triangulate=FALSE,compile=FALSE)
cgtcaca<-gt2aca(mixMBN,Cgt)
add.child.meiosis.nodes(mixMBN,cgtcaca,1)
log10LR<-(logLX(mixMBN,
expr.make.findings(list(
list('Male',ind=1),
list('Caca',aca='cgtcaca')
))
)(pars)-attr(cgtcaca,'logGt')-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

`loop.rpt.IBD` *Analysis of DNA mixtures with familial relationships by looping*

Description

Analysis of DNA mixtures with familial relationships, by looping over traces, markers, and IBD patterns, to reduce total BN table size, at some price in execution time

Usage

```
loop.rpt.IBD(listdata, pars, IBD, typed.gts = NULL, inds = 1,
    jtyped = ncol(IBD$patt)/2 - length(typed.gts) + seq_along(typed.gts),
    jcontr = seq_along(inds), targets = NULL, contribs,
    quiet=FALSE, verbose=FALSE, presence.only=FALSE, ...)
```

Arguments

`listdata`	as in call to DNAmixture
`pars`	parameter structure, in `mixpar` format
`IBD`	multi-person coefficients of identity, in any of the formats accepted by `convertIBD`
`typed.gts, inds, jtyped, jcontr, targets, contribs, quiet`	as in call to rpt.IBD
`verbose`	should per-marker and overall log10LR's be reported?
`presence.only`	Set to TRUE to ignore peak heights and evaluate the likelihood function considering peak presence and absence (heights above and below threshold) only. Defaults to FALSE.
`...`	other arguments to DNAmixture, particularly including `k`, `C`, `database`

Value

The value of the overall `log10 LR`, and the contributions of individual markers in the form of a vector-valued attribute 'log10LR', are returned invisibly; individual marker/pattern values are also printed out.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

C<-0.001

## Fit 3-person mixture - baseline model

mixD<-DNAmixture(list(epg),k=3,C=rep(list(C),length(list(epg))),database=db)
pars3<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.6,U2=0.3,U3=0.1)))
baseline3<-logL(mixD)(pars3)
size(mixD)

## Fit 3-person mixture - in which U1 and U2 have a parent-child relationship

mixD<-DNAmixture(list(epg),k=3,C=rep(list(C),length(list(epg))),database=db,
triangulate=FALSE,compile=FALSE)
delete.DQnodes(mixD)
rpt.IBD(mixD,IBD=c(0,1,0),typed.gts=list(),inds=1:2,jtyped=NULL)
size(mixD)
log10LR<-(logL(mixD)(pars3)-baseline3)/log(10)
cat('log10 LR',log10LR,'\n')

## the same analysis by loop.rpt.IBD

listdata<-list(epg)
print(loop.rpt.IBD(listdata,pars3,IBD=c(0,1,0),
k=3,C=rep(list(C),length(listdata)),database=db,
typed.gts=list(),inds=1:2,jtyped=NULL))
```

---

make.profile *Convert genotype profile to reference profile format*

Description

Convert genotype profile to reference profile format

Usage

```
make.profile(gt,name='K')
```

Arguments

gt	genotype profile
name	character string used to name profile in output data frame

Value

data frame containing reference profile

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
S1prof<-make.profile(S1gt,'S1')
C<-0.001
mixD<-DNAmixture(list(epg),k=3,K='S1',reference.profile=S1prof,C=list(C),database=db)
```

`mixMLX` *Maximises mixture likelihood when extra findings present*

Description

Replacement for mixML in DNAmixtures that calls logLX instead of logL.

Usage

```
mixMLX(mixture, expr.extra.findings, pars, constraints = NULL, phi.eq = FALSE,
    val = NULL, trace = FALSE, order.unknowns = TRUE, initialize = FALSE, ...)
```

Arguments

`mixture`	Compiled DNAmixture object.
`expr.extra.findings`	expression containing the extra findings
`pars`	Parameters, in `mixpar` format.
`constraints`	as in `mixML`
`phi.eq`	as in `mixML`
`val`	as in `mixML`
`trace`	as in `mixML`
`order.unknowns`	as in `mixML`
`initialize`	should all entered evidence be removed from the networks in `mixture`
`...`	as in `mixML`

Value

A list containing

mle The (suggested) MLE.

lik The log of the likelihood (log e).

as well as the output from the optimisation.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

See Also

See also `mixML`.

Examples

```
data(test2data)

# set threshold C
C<-0.001

mixD<-DNAmixture(list(epg),k=2,C=list(C),database=db)

# find MLE's and maximised likelihood
# adding evidence individual 1 is Male

expr.extra.findings<-expr.make.findings(list(list('Male',ind=1)))

startpar<-mixpar(rho=list(60),eta=list(24),xi=list(0.16),phi=list(c(U1=0.75,U2=0.25)))
mlDM<-mixMLX(mixD,expr.extra.findings,startpar,trace=FALSE)
pars<-mlDM$mle
cat('\nBaseline model maximised likelihood:',mlDM$lik,'\n')
cat('and MLEs:\n')
print(mlDM$mle)
```

pedigreeIBD — *Construct IBD pattern distribution from pedigree*

Description

Construct IBD pattern distribution from a pedigree and a target list of individuals

Usage

```
pedigreeIBD(x, targets, cond = TRUE, ped=FALSE, quiet = TRUE, verbose = FALSE)
```

Arguments

`x`	A pedigree in `pedtools` format
`targets`	Character vector, some or all of the individual identifiers in the pedigree `x`
`cond`	should IBD pattern be condensed?
`ped`	logical, should complete pedigree be added as an attribute to the output, if available?
`quiet`	should resulting IBD pattern distribution be printed?
`verbose`	should trace information be printed?

Details

This function computes the multi-person condensed coefficients of identity for an arbitrary set of individuals, in the sparse notation of the IBD pattern distribution of Green and Vigeland (2019).

Value

IBD pattern distribution as a list with components `pr` and `patt`

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

References

Multi-person condensed coefficients of identity, by Peter J. Green and Magnus Dehli Vigeland, University of Bristol technical report, 2019.

See Also

`pedtools`, `formats`

Examples

```
require(ribd)

id<-c('gf','gm','b1','b2','m','c')
fid<-c(0,0,'gf','gf',0,'b1')
mid<-c(0,0,'gm','gm',0,'m')
sex<-c(1,2,1,1,2,0)
x<-ped(id,fid,mid,sex)

IBD<-pedigreeIBD(x,c('m','c','b1','b2'))

kappaIBD(x,c('m','c','b1','b2'))
```

`plot.IBD` *Plot IBD patterns and pattern distributions*

Description

Plot IBD patterns and pattern distributions

Usage

```
## S3 method for class 'IBD'
plot(x,labels=NULL,probs=NULL,order=NULL,colrs=c('black','red','blue'),
digits=3,nr=ceiling(sqrt(np)),...)
```

Arguments

`x`	A matrix whose rows are IBD patterns, or a list whose components are `patt`, such a matrix, together with `pr`, a vector of the corresponding probabilities
`labels`	Vector of numerical or character labels for the patterns, if `NA`, labels are constructed from the patterns by catenation, if `NULL`, the labels are not displayed.
`probs`	Vector of probabilities of the patterns, if not provided as a component of `pattern`; if `NULL`, the probabilities are not displayed.
`order`	A character string, partially matched using `pmatch` to one of 'pattern', 'probs', or 'labels', requesting ordering diagram accordingly (in the case of `probs` in decreasing order, **or** a numeric, complex, character or logical vector of length the number of patterns, requesting ordering by this variable, **or** NULL (the default), requesting no re-ordering.
`colrs`	A vector of colours: ties in the ordering variable are indicated by coloured groups, with colours chosen cyclically from this vector.
`digits`	Integer, overwriting default number of significant digits for `probs`
`nr`	Integer, overwriting default number of rows for plotted array, default a rounding up of the square root of the number of patterns.
`...`	additional arguments to `plot`

Value

No value is returned, the function is called for its side effect, a plot on the current display device.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
require(ribd)
data(emperors)

plot.IBD(convertIBD('3cousins-star'),order='probs',col=c('blue','red','black'))

plot(attr(emperors,'ped'))

o<-order(emperors$pr)[1:12]
plot.IBD(emperors$patt[o,],probs=emperors$pr[o],labels=NA,order='probs')
```

`protected` *Catch numerical errors, and return -Inf*

Description

Attempts to catch numerical erros in evaluating the expression `x`, delivering a default result instead of NaN or other failures

Usage

```
protected(x,default=-Inf)
```

Arguments

`x`	expression to be evaluated, typically the log-likelihood of a modified mixture model
`default`	value to be delivered if numerical errors are encountered

Value

Returns -Inf in case of error, otherwise the value of `x`

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

`protected.mixML` *Protect against numerical errors in maximum likelihood computation*

Description

Attempts to catch numerical errors in maximum likelihood computation, by replacing logL values by a default value instead of NaN or other failures

Usage

```
protected.mixML(mixture, pars, constraints = NULL, phi.eq = FALSE, val = NULL,
  trace = FALSE, order.unknowns = TRUE, default=-999999, ...)
```

Arguments

`mixture`	A DNAmixture object.
`pars`	A mixpar parameter used as a starting value for the optimisation.
`constraints`	Equality constraint function as function of an array of parameters.
`phi.eq`	Should the mixture proportions be the same for all traces? Defaults to FALSE.
`val`	Vector of values to be satisfied for the equality constraints.
`trace`	Print the evaluations of the likelihood-function during optimisation?
`order.unknowns`	Should unknown contributors be ordered according to decreasing contributions? Defaults to TRUE.
`...`	Further arguments to be passed on to solnp.
`default`	value of logL to be used if numerical errors are encountered

Value

A list containing

`mle`	The (suggested) MLE.
`lik`	The log of the likelihood (log e).

as well as the output from the optimisation.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

`require.compiled` *Force compilation of all BNs in a DNA mixture model*

Description

Scan all Bayes nets in mixture, and compile any that are not already compiled

Usage

```
require.compiled(mixture)
```

Arguments

`mixture`	A DNAmixture object

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,dyes=list(NGMDyes),
triangulate=FALSE,compile=FALSE)
replace.tables.for.UAF(mixD,40,compile=FALSE)
require.compiled(mixD)
```

`rGTs` *Simulate random genotype profiles and DNA samples for related individuals*

Description

Simulate random genotype profiles and DNA samples for arbitrarily related individuals

Usage

```
rGTs(nreps,IBD,db,DNA,sex=rep(0,ncontr),nU=0)
```

Arguments

`nreps`	Integer, number of replicates
`IBD`	Specification of relationships, as in `convertIBD`
`db`	Data frame, database of alleles and their frequencies, for each marker; for format, see `formats`.
`DNA`	Integer vector, numbers of DNA cells for the respective individuals, can be NA
`sex`	Integer vector, sex of the respective contributors: 1=male, 2=female, 0=unspecified
`nU`	Integer, include also this number of unrelated individuals

Details

Genotype profiles are generated randomly, using the allele frequency database `db`, under the relationships specified by the `IBD` argument. In accordance with the underlying biology, allele values for the AMEL marker (if this is one of the markers included) are not influenced by relationships with other individuals; however they are influenced by the sex of the individuals, where this is known. Information on sex can be specified by the optional argument `sex`: a male is given the profile X-Y, a female X-X, and an individual with unspecified sex X-X or X-Y with equal probabilities.

Value

Data frame with variables `Sim`, `Sample.name`, `Marker`, `Allele`, and `DNA`, suitable for input to `simExtraction`, etc. See package `pcrsim`.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)
```

`rni` *Random number initialiser supporting spontaneous replication*

Description

Random number initialiser supporting spontaneous replication

Usage

```
rni(seed=0)
```

Arguments

`seed`	Integer, seed

Details

This is a convenience front end to `set.seed`. A non-zero value of `seed` is passed directly to `set.seed`. Given a zero value (the default), the function calls `Sys.time` to generate an unpredictable starting value – but the value ultimately passed to `set.seed` is both output using `cat` and returned invisibly, so can be used for unanticipated replica runs of a simulation.

Value

Non-zero seed value that can be used to reproduce run subsequently

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
rni(0)
runif(6)
rni(0)
runif(6)
rni(3456)
runif(6)
rni(3456)
runif(6)
keep<-rni(0)
print(keep)
runif(6)
rni(keep)
runif(6)
```

`rpt.AMEL` *Replace CPTs for AMEL marker in a DNA mixture*

Description

Used after a call to `DNAmixture` with `compile=FALSE,triangulate=FALSE`, this function replaces the CPTs for the genotype allele count arrays for the AMEL marker in a DNA mixture to specify sex of contributors

Usage

```
rpt.AMEL(mixture,sex,compile=TRUE)
```

Arguments

`mixture`	A DNAmixture object
`sex`	Integer vector, sex of each contributor
`compile`	Logical, should BN be compiled after modification?

Details

The sex of each contributor is coded as in `pedtools`, namely 0=unspecified, 1=male, 2=female.

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

mixD<-DNAmixture(list(epg),k=3,C=list(0.001),database=db,dyes=list(NGMDyes),
        triangulate=FALSE,compile=FALSE)
rpt.AMEL(mixD,c(1,2,0)) # the 3 contributors are male, female, and unspecified,respectively.
```

---

rpt.IBD *Replacing CPTs for selected mixture contributors with familial relationships*

Description

Used after a call to DNAmixture with compile=FALSE,triangulate=FALSE, this function replaces the CPTs for the genotype allele count arrays for specified mixture contributors by those representing the specified relationship with each other and typed relatives

Usage

```
rpt.IBD(mixture, IBD="parent-child", typed.gts = NULL, inds = 1,
jtyped = ncol(IBD$patt)/2 - length(typed.gts) + seq_along(typed.gts),
jcontr = seq_along(inds),
targets=attr(IBD,'targets'), contribs=NULL, quiet=FALSE, all.freq = NULL, compile = TRUE)

rpt.typed.relatives(mixture, IBD="parent-child", typed.gts = NULL, inds = 1,
jtyped = ncol(IBD$patt)/2 - length(typed.gts) + seq_along(typed.gts),
jcontr = seq_along(inds),
targets=attr(IBD,'targets'), contribs=NULL, quiet=FALSE, all.freq = NULL, compile = TRUE)

rpt.typed.child(mixture, aca, ind=1)

replace.Ui.tables(mixture, aca, ind=1)

rpt.typed.parents(mixture, Mgt, Fgt, ind=1, compile=TRUE)

rpt.typed.relative(mixture, Rgt, IBD=c(0.25,0.5,0.25), ind=1, compile=TRUE)
```

Arguments

mixture	DNAmixtures object created by previous call to DNAmixture with triangulate=FALSE,compile=FALSE
IBD	relationships between the specified individuals, as multi-person condensed coefficients of identity, in one of several representation; see Details.
typed.gts	list of 0 or more genotypes of relatives; the components of this list must be named (with the id's of the relevant individuals) if targets and contribs are used to code the correspondences (see Details).

`inds`	vector of 1 or more integers: which ‘unknown’ contributors are we modelling by amending their CPTs? The elements should be listed in the same order as the corresponding pairs of columns of the IBD patterns in `IBD`
`jtyped`	indices of pairs of columns of `IBD$patt` that correspond to the typed relatives (if any); default the last `length(typed.gts)` pairs of columns
`jcontr`	indices of pairs of columns of `IBD$patt` that correspond to the relevant mixture contributors; default the first `length(inds)` pairs of columns
`targets`	Character vector of the tags of the individuals referred to in `IBD`
`contribs`	Character vector of the tags of the individuals included in the mixture, in order
`quiet`	should calculated values of inds, jtyped and jcontr be reported?
`all.freq`	alternative allele frequency database(s), see Details.
`compile`	logical flag: should mixture object be compiled on exit?
`ind`	as `inds`, used above when only one allowed
`aca, Mgt, Fgt, Rgt`	individual genotypes, as allele count arrays

Details

In using `rpt.IBD` or `rpt.typed.relatives` (which is identical), the correspondence between mixture contributors, specified relationships, and typed genotype profiles should be specified **either** (preferably) using `targets`, `contribs` and through the names of the components in `typed.gts`, **or** (to be deprecated) with `inds`, `jcontr` and `jtyped`: the two representations should not be mixed up. If either `targets` or `contribs` specified, the former representation is assumed.

Special cases are treated slightly more efficiently: `rpt.typed.child`: single contributor, single typed relative, parent or child; `rpt.typed.parents`: single contributor, both parents typed; `rpt.typed.relative`: single contributor, single typed relative.

Note that `IBD$patt` always has an even number of columns, two for each individual in the joint relationship specified; `jtyped` and `jcontr` are vectors of indices of these individuals, i.e. to pairs of adjacent columns of `IBD$patt`.

Multiple functions in this group can validly be called sequentially (with all but the last having `compile=FALSE`) providing they reference different sets of contributors among the targets, **and** that these sets are conditionally independent given the typed genotypes specified.

There are multiple valid representations for relationships in the argument `IBD` – as an IBD pattern distribution, via a pedigree, or. in the case of just two individuals. via either a vector of 3 kappas or 9 Deltas (Jacquard’s condensed coefficients of identity). For full details, see `convertIBD`. If `IBD` is missing, the default value represents parent-child.

In the interests of upward compatibility, in `rpt.typed.child` and `replace.Ui.tables` (which are identical), the argument `Cgt` can be given as either a genotype profile data frame, or an allele count array.

By default, the allele frequency database used for the founding genes is that used when the `mixture` object is created, in an earlier call to `DNAmixture`. A non-null value for the `all.freq` argument allows the user to specify alternative database (s) for the founding genes. If its value is an allele frequency database (in the format specified in `formats`) then that database is used for all founding genes; if the value of the argument is a list of such databases, then component k of the list is used for allele frequencies for the founding gene labelled k in the IBD argument. Note that this option allows modelling of mixtures where different contributors are drawn from different populations, whether or not there are relationships among individuals.

Value

Vector of marker-specific probabilities of the typed genotypes.

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

## Fit 2-person mixture - baseline model

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

## Fit 2-person mixture model in which contributor 1 is parent of a typed individual Cgt

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,triangulate=FALSE,compile=FALSE)
rpt.IBD(mixD,,list(Cgt))
log10LR<-(logL(mixD)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')

## Fit 2-person mixture model in which contributor 1 is father of a typed individual Cgt
## with mother Mgt

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,triangulate=FALSE,compile=FALSE)
rpt.IBD(mixD,,list(Mgt,Cgt))
log10LR<-(logL(mixD)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')

## Fit 2-person mixture model in which contributors are two parents of a child with
## genotype Cgt, and a parent of one of them has genotype Rgt. Note the encoding of allele
## labels to reduce the complexity of the IBD pattern distribution IBD.

IBD<-list(patt=rbind(c(1,3,2,4,1,2,1,5),c(1,3,2,4,1,2,3,5)))

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,triangulate=FALSE,compile=FALSE)
rpt.IBD(mixD,IBD,list(Cgt,Rgt),1:2)
log10LR<-(logL(mixD)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')

## the same, with individuals and relationships denoted by character tags

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,triangulate=FALSE,compile=FALSE)
```

```
rpt.IBD(mixD,IBD,list(c=Cgt,gf=Rgt),targets=c('f','m','c','gf'),contribs=c('f','m'))
log10LR<-(logL(mixD)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

`rpt.UAF` *Replace CPTs in a DNA mixture to model uncertain allele frequencies*

Description

Replace CPTs in a DNA mixture to model uncertainty in allele frequencies

Usage

```
replace.tables.for.UAF(mixture, M, compile = TRUE)

rpt.UAF(mixture, M, compile = TRUE)
```

Arguments

`mixture`	DNAmixtures object created by previous call to `DNAmixture` with `triangulate=FALSE,compile=FALSE`
`M`	Size of allele frequency database
`compile`	logical flag: should mixture object be compiled on exit?

Value

No value is returned, the function is called for its side effect

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

## Fit 2-person mixture - baseline model

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db)
pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.7,U2=0.3)))
baseline<-logL(mixD)(pars)

## Fit 2-person mixture model under assumption that database size was only 40

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db,triangulate=FALSE,compile=FALSE)
replace.tables.for.UAF(mixD,40)
log10LR<-(logL(mixD)(pars)-baseline)/log(10)
cat('log10 LR',log10LR,'\n')
```

size *Calculate and display total size of BN tables for a DNA mixture*

Description

Calculate and display total size of BN tables for a DNA mixture

Usage

```
size(mixture)
```

Arguments

`mixture`	A compiled DNAmixture object

Value

Returns total size, typically to be printed by bespoke method

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

Examples

```
data(test2data)
data(NGMDyes)

## Fit 2-person mixture - baseline model

mixD<-DNAmixture(list(epg),k=2,C=list(0.001),database=db)
size(mixD)
```

test2data *Small data set for demonstrating some capabilities of KinMix and KinMixLite*

Description

Small test data set (2 markers with 4 or 5 alleles each, plus AMEL), for demonstrating some capabilities of KinMix and KinMixLite

Usage

```
data("test2data")
```

Format

Data objects for demonstrating KinMix: `epg` (DNAmixtures peak height data), `db` (DNAmixtures allele frequency database), and `Cgt`, `Fgt`, `Mgt`, `Rgt`, `S1gt`, `S2gt` potential relative genotype data frames.

Examples

```
data(test2data)
```

`wlr` *Computes paternity LR using WLR method*

Description

Computes overall LR from Ugt-specific LR's using estimated Ugt genotype profile in `sep` corresponding to contributor i in the mixture as Father; uses Child genotype information in `Cgt` data.frame and optionally Mother's genotype in `Mgt`. Implements method WLR.

Usage

```
wlr(sep, Cgt, db, ind=1, Mgt=NULL)
```

Arguments

`sep`	Separation, a list of configurations of genotypes for some or all unknown contributors, output by `map.genotypes`.
`Cgt`	Child's genotype profile as a data frame containing variables `marker`, `allele1` and `allele2`
`db`	Allele frequency database
`ind`	Index of contributor regarded as Father
`Mgt`	(optionally) Mother's genotype profile as a data frame containing variables `marker`, `allele1` and `allele2`

Value

Returns LR for paternity

Author(s)

Peter Green (P.J.Green@bristol.ac.uk)

See Also

See also `map.genotypes`.

Examples

```
data(test2data)
data(NGMDyes)

# set threshold C
C<-0.001

pars<-mixpar(rho=list(2),eta=list(100),xi=list(0.1),phi=list(c(U1=0.9,U2=0.1)))

mixWLR<-DNAmixture(list(epg),k=2,C=list(C),database=db,dyes=list(NGMDyes))
setPeakInfo(mixWLR,pars)
sepWLR<-map.genotypes(mixWLR,type="all",pmin=0.0001,U=1)
LR<-wlr(sepWLR,Cgt,db)
cat('\nWLR LR:',LR,'; log10(LR):',log10(LR),'\n')
```

Index