Package 'relMix'

December 14, 2020

Type Package

Title Relationship Inference for DNA Mixtures

Version 1.3.3

Date 2020-12-14

Description Makes relationship inference involving DNA mixtures with unknown profiles.

VignetteBuilder knitr

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Familias, gWidgets2, gWidgets2tcltk

Suggests knitr, pander, rmarkdown, tkrplot

License GPL (>= 2)

RoxygenNote 7.1.1

NeedsCompilation no

Author Guro Dorum [aut],
Elias Hernandis [ctb, cre],
Navreet Kaur [ctb],
Thore Egeland [ctb]

Maintainer Elias Hernandis <eliashernandis@gmail.com>

Repository CRAN

Date/Publication 2020-12-14 16:30:03 UTC

R topics documented:

`allGenos` *Find all possible genotypes*

Description

Finds all possible genotypes based on input alleles.

Usage

```
allGenos(alleles)
```

Arguments

`alleles`	Vector of input alleles, numeric or character

Value

Matrix of all possible genotypes, one row per genotype

Author(s)

Guro Dorum

`checkFrequenciesFile` *Load and check a frequency file*

Description

Loads a frequency database file and compares it against mixture data to check for common errors.

Usage

```
checkFrequenciesFile(filename, mix)
```

Arguments

`filename`	Path of the frecuency database file
`mix`	Data frame with mixture data. See relMix vignette for description of the format

Details

The mixture data is used to perform more advanced checks, such as to make sure all alleles present in the mixture file have an entry in the frequency database. If warnings are found, the function attempts to fix them and explains what it has done in the warning messages. If an error is found, checking stops and a NULL dataframe is returned. The error is described in the error messages.

Value

A list containing

- `df` Data frame with frequencies
- `warning` List of strings describing the errors that ocurred but could be fixed or that do not prevent the execution of the program.
- `error` List of strings describing the errors that ocurred that made it imposible to return a valid data frame. If this list is not empty, then the dataframe item will be NULL

Author(s)

Elias Hernandis

See Also

`checkMixtureFile` for information on how to load a mixture file.

Examples

```
## Not run:
mixfile <- system.file("extdata","mixture.txt",package="relMix")
mix <- checkMixtureFile(mixfile)
# note: the mixture dataframe is passed as an argument
# if the previous check failed, the program should not continue
# with the frequencies file check
freqfile <- system.file('extdata','frequencies22Markers.txt',package='relMix')
freqs <- checkFrequenciesFile(freqfile, mix$df)

## End(Not run)
```

`checkMixtureFile` *Load and check a mixture file*

Description

Given a mixture file name, returns the loaded data frame along with any detected errors or warnings.

Usage

```
checkMixtureFile(filename)
```

Arguments

filename	Path of the mixture file

Details

If warnings are found, the function attempts to fix them and explains what it has done in the warning messages. If an error is found, checking stops and a NULL dataframe is returned. The error is described in the error messages.

Value

A list containing

- df The loaded data frame, NULL if errors are present.
- warning A list of strings describing the errors that ocurred but could be fixed or that do not prevent the execution of the program.
- error A list of strings describing the errors that ocurred that made it imposible to return a valid data frame. If this list is not empty, then the dataframe item will be null.

Author(s)

Elias Hernandis

Examples

```
## Not run:
mixfile <- system.file("extdata","mixture.txt",package="relMix")
result <- checkMixtureFile(mixfile);
print(result$df);
print(result$warning);
print(result$error);

## End(Not run)
```

---

checkPedigreeFile *Check a pedigree file*

Description

Given a pedigree file path the function attempts to load it and compare it to the reference profiles to detect possible errors.

Usage

```
checkPedigreeFile(filename, df)
```

Arguments

filename	Path of the pedigree file
df	Data frame with reference profiles

Details

The pedigree file must be a .R file defining a pedigree (see the relMix vignette for an example). The data frame with reference data is used to compare names of individuals and detect possible misspellings. If warnings are found, the function attempts to fix them and explains what it has done in the warning messages. If an error is found, checking stops. The error is described in the error messages.

Value

A list containing

- df Pedigree, NULL if errors are present.
- warning A list of strings describing the errors that ocurred but could be fixed or that do not prevent the execution of the program.
- error A list of strings describing the errors that ocurred that made it imposible to return a valid data frame. If this list is not empty, then the dataframe item will be null.

Author(s)

Elias Hernandis

Examples

```
## Not run:
#First load mixture file
mixfile <- system.file("extdata","mixture_silent_ex.txt",package="relMix")
mix <- checkMixtureFile(mixfile);
#Load reference file
reffile <- system.file("extdata","references_silent.txt",package="relMix")
ref <- checkReferenceFile(reffile, mix$df)
#Check pedigree file
pedfile <- system.file("extdata","custom_pedigree_maternity_duo.R",package="relMix")
checkPedigreeFile(pedfile,ref$df);

## End(Not run)
```

`checkReferenceFile` *Check a reference profiles file*

Description

Given a reference profile file name the function attempts to load it and compare it to the mixture file to detect possible errors.

Usage

```
checkReferenceFile(filename, mix)
```

Arguments

`filename`	Path of the reference profiles file
`mix`	Data frame with mixture data

Details

See the relMix vignette for a description of the format of the reference file. The data frame with mixture data is used to compare If warnings are found, the function attempts to fix them and explains what it has done in the warning messages. If an error is found, checking stops and a NULL dataframe is returned. The error is described in the error messages.

Value

A list containing

- `df` The loaded data frame, NULL if errors are present
- `warning` A list of strings describing the errors that ocurred but could be fixed or that do not prevent the execution of the program.
- `error` A list of strings describing the errors that ocurred that made it impossible to return a valid data frame. If this list is not empty, then the data frame item will be null.

Author(s)

Elias Hernandis

See Also

`checkMixtureFile` for information on how to load a mixture file.

Examples

```
## Not run:
#Load a mixture file
mixfile <- system.file("extdata","mixture.txt",package="relMix")
mix <- checkMixtureFile(mixfile);
#Note: the mixture dataframe is passed as an argument. If the previous check failed,
#the program should not continue with the reference file check
reffile <- system.file("extdata","references.txt",package="relMix")
checkReferenceFile(reffile, mix$df);

## End(Not run)
```

`createDatamatrix` *Create data matrix with possible genotype combinations for specified individuals*

Description

A data matrix of genotypes for known individuals and all possible genotypes for unknown individuals is created.

Usage

```
createDatamatrix(locus, knownGenos, idsU = NULL)
```

Arguments

`locus`	A Familias locus containing information about the alleles
`knownGenos`	List of known genotypes. Each element is a vector with genotype for one individual. The elements must be named
`idsU`	Vector of indices for unknown individuals

Value

A data matrix of genotypes where each row corresponds to an individual.

Author(s)

Guro Dorum

See Also

`FamiliasLocus` and `relMix`.

Examples

```
#Define alleles and frequencies
alleles <- 1:2
afreq <- c(0.5,0.5)
#Create locus object
locus <- Familias::FamiliasLocus(frequencies=afreq,name="M1",allelenames= alleles)
#Known genotypes of alleged father and mother, child's genotype is uknown
gAF <- c(1,1)
gMO <- c(1,1)
datamatrix <- createDatamatrix(locus,knownGenos=list(AF=gAF,MO=gMO),idsU=c("CH"))
```

---

db *Allele database*

Description

Norwegian database with 17 EXS17 markers and 6 additional markers.

Usage

```
data(db)
```

Format

A data frame with 324 observations on the following 3 variables:

`Marker` a factor with levels corresponding to name of markers

`Allel` a numeric vector denoting allele

`Frequency` a numeric vector in (0,1)

Source

Dupuy et al. (2013), unpublished.

Examples

```
data(db)
#Checks that frequencies add to 1
lapply(split(db$Frequency,db$Marker),sum)
#Finds number of alleles for all markers
unlist(lapply(split(db$Frequency,db$Marker),length))
#A closer look at the marker SE33
SE33=db[db$Marker=="SE33",]
barplot(SE33$Frequency)
```

db2 *Allele database for 22 markers*

Description

Frequencies for 22 loci from the prototype 24-plex STR panel from Thermo Fisher.

Usage

```
data(db2)
```

Format

A data frame with 206 observations on the following 3 variables.

`Marker` a factor with levels corresponding to name of markers

`Allele` a numeric vector denoting allele

`Frequency` a numeric vector in (0,1)

Details

The format is convenient for R.

Source

Hill et al. (2013) U.S. population data for 29 autosomal STR loci. Forensic Sci. Int. Genet. 7, e82-e83.
Hill et al. (2006) Allele Frequencies for 26 MiniSTR Loci with U.S. Caucasian, African American, and Hispanic Populations. http://www.cstl.nist.gov/biotech/strbase/NISTpop.htm

Examples

```
data(db2)
```

generateMix *Create a mixture of genotypes with simulated drop-in and dropout*

Description

Takes as input genotypes and creates a mixture. Alleles drop in and out of the mixture with the specified probabilities

Usage

```
generateMix(G, alleles, afreq, D, di)
```

Arguments

G	List of genotypes. Each element is a vector with genotype for one individual
alleles	Numeric or character Vector of allele names for the marker
afreq	Numeric vector of allele frequencies for the marker
D	List of dropout values (between 0 and 1) per contributor. Each element is a vector containing heterozygous and homozygous dropout probability for the given contributor
di	Drop-in value (between 0 and 1)

Value

A vector of mixture alleles.

Author(s)

Guro Dorum

Examples

```
#Define alleles and frequencies
alleles <- 1:2
afreq <- c(0.5,0.5)
#Genotypes
gM <- c(1,1)
gC <- c(1,2)
#Dropout and drop-in values
d <- 0.1
di <- 0.05
#No drop-in for first contributor
D <- list(c(0,0),c(d,d^2))
R <- generateMix(G=list(gM,gC),alleles,afreq,D=D,di=di)
```

---

mixLikDrop *Likelihoods for DNA mixtures with dropout and dropin*

Description

Computes the likelihood of a mixture conditioned on a given number of known and unknown contributors, and drop-in and dropout probabilities.

Usage

```
mixLikDrop(R, G, D, di = 0, alleleNames, afreq)
```

Arguments

R	Vector of mixture alleles
G	List of genotypes. Each element is a vector with genotype for one individual
D	List of dropout values (between 0 and 1) per contributor. Each element is a vector containing heterozygous and homozygous dropout probability for the given contributor
di	Drop-in value (between 0 and 1)
alleleNames	Vector of allele names for the marker
afreq	Vector of allele frequencies for the marker

Value

The likelihood

Author(s)

Guro Dorum

References

The model is specified in the appendix of Haned et al. (2012) Exploratory data analysis for the interpretation of low template DNA mixtures, FSI Genetics Dec;6(6):762-74

See Also

relMix

Examples

```
#Define alleles and frequencies
alleles <- 1:2
afreq <- c(0.5,0.5)
#Genotypes
gM <- c(1,1)
gC <- c(1,2)
#Mixture alleles
R <- c(1,2)
#Dropout and drop-in values
d <- 0.1
di <- 0.05
#No drop-in for first contributor
D <- list(c(0,0),c(d,d^2))
mixLikDrop(R=R,G=list(gM,gC),D=D,di=di,alleleNames=alleles,afreq=afreq)
```

relMix *Relationship inference based on mixtures*

Description

Calculates likelihoods for relationship inference involving mixtures and missing reference profiles, including drop-in and dropout, mutations, silent alleles and theta correction.

Usage

```
relMix(
  pedigrees,
  locus,
  R,
  datamatrix,
  ids,
  D = rep(list(c(0, 0)), length(ids)),
  di = 0,
  kinship = 0
)
```

Arguments

pedigrees	A list of pedigrees defined using FamiliasPedigree in Familias
locus	A Familias locus. Note that a silent allele must be indicated by 's' (and not 'silent' as in Familias)
R	A vector of mixture alleles, or a list of such if there are multiple replicates
datamatrix	Each line corresponds to one constellation of genotypes for the involved individuals. Indices of individuals must be given as rownames and must correspond to indices in the pedigree
ids	Index vector indicating which individuals are contributors to the mixture. The indices must correspond to indices in the pedigree
D	List of dropout values (between 0 and 1) per contributor. Each element is a vector containing heterozygous and homozygous dropout probability for the given contributor
di	Drop-in value (between 0 and 1)
kinship	Defines the theta-parameter

Details

The function requires the package **Familias** and calls on the function `FamiliasPedigree`.

Value

The likelihoods for the pedigrees and detailed output for each term considered in the calculation.

Author(s)

Navreet Kaur, Thore Egeland, Guro Dorum

References

Dorum et al. (2017) Pedigree-based relationship inference from complex DNA mixtures, Int J Legal Med., doi:10.1007/s00414-016-1526-x;
Kaur et al. (2016) Relationship inference based on DNA mixtures, Int J Legal Med.;130(2):323-9;
Egeland, Kling, Mostad (2015) **Familias**

See Also

`relMixGUI` for a relMix GUI, and `FamiliasLocus` on how to create a Familias locus.

Examples

```
#Example 1: paternity trio with mixture of mother and child
#Define alleles and frequencies
alleles <- 1:2
afreq <- c(0.4,0.6)
#Define pedigrees
persons <- c("CH","MO","AF")
ped1 <- Familias::FamiliasPedigree(id=persons, dadid=c("AF",NA, NA), momid=c("MO", NA,NA),
                        sex=c("male", "female", "male"))
ped2 <- Familias::FamiliasPedigree(id=c(persons, "TF"), dadid=c("TF", NA, NA,NA),
                      momid=c("MO", NA, NA,NA), sex=c("male", "female", "male", "male"))
pedigrees <- list(isFather = ped1, unrelated=ped2)
#Create locus object
locus <- Familias::FamiliasLocus(frequencies=afreq,name="M1",
                      allelenames= alleles)
#Known genotypes of alleged father and mother
gAF <- c(1,1)
gMO <- c(1,1)
#Mixture alleles
R <- c(1,2)
datamatrix <- createDatamatrix(locus,knownGenos=list(AF=gAF,MO=gMO),idsU=c("CH"))
#Define dropout and drop-in values
d <- 0.1
di <- 0.05
res <- relMix(pedigrees, locus, R, datamatrix, ids=c("MO","CH"),
            D=list(c(0,0),c(d,d^2)),di=di, kinship=0)
#LR=0.054
res$isFather/res$unrelated

#Example 2: Exhaustive example with silent allele, mutations, dropout and drop-in
#H1: Contributors are mother and child
#H2: Contributors are mother and unrelated
#Possible dropout in both contributors
gMO <- c(1,1) #Mother's genotype
R <- 1 #Mixture alleles
#Mother/child pedigree
persons <- c("CH","MO")
```

```
ped1 <- Familias::FamiliasPedigree(id=persons, dadid=c(NA,NA), momid=c("MO", NA),
                        sex=c("male", "female"))
ped2 <- Familias::FamiliasPedigree(id=c(persons), dadid=c(NA, NA),
                        momid=c( NA, NA),
                        sex=c("male", "female"))
pedigrees <- list(H1 = ped1, H2=ped2)
#Alleles and frequencies:
#When silent alleles are involved, a custom mutation matrix is required.
#No mutations are possible to or from silent alleles.
#We create the mutation model with FamiliasLocus and modify it before it is
#passed on to relMix
alleles <- c(1,2,'silent')
afreq <- c(0.4,0.5,0.1)
#Create initial locus object with mutation matrix
locus <- Familias::FamiliasLocus(frequencies=afreq,name='M1',
                      allelenames= alleles, MutationModel='Equal',
                      femaleMutationRate=0.1,maleMutationRate=0.1)
#Modify mutation matrix from Familias:
#Silent allele must be given as 's' (not 'silent' as in Familias)
newAlleles <- c(alleles[-length(alleles)],'s')
mm <- locus$femaleMutationMatrix
colnames(mm) <- rownames(mm) <- newAlleles
#Create new locus object with modified mutation matrix
locus <- Familias::FamiliasLocus(frequencies=afreq,name='M1',
                     allelenames= newAlleles, MutationModel='Custom', MutationMatrix=mm)
knownGenos <- list(gMO)
names(knownGenos) <- c("MO")
datamatrix <- createDatamatrix(locus,knownGenos,ids="CH")
d <- 0.1 #Dropout probability for both contributors
di <- 0.05
res2 <- relMix(pedigrees, locus, R, datamatrix, ids=c("MO","CH"),
              D=list(c(d,d^2),c(d,d^2)),di=di, kinship=0)
#LR=1.68
res2$H1/res2$H2
```

`relMixGUI` *GUI for relMix*

Description

User-friendly graphical user interface for relMix.

Usage

```
relMixGUI()
```

Details

Includes error checking for the input files.

Author(s)

Guro Dorum, Elias Hernandis

See Also

relMix for the main function implemented in relMixGUI.

Examples

```
#Examples can be found in the vignette and example data files can be found
#in the folder "inst\extdata" in the installation folder for relMix
```

Index