Package ‘numberofalleles’

October 13, 2022

Type Package

Title Compute the Probability Distribution of the Number of Alleles in
a DNA Mixture

Version 1.0.1

Date 2022-04-28

Description The number of distinct alleles observed in a DNA mixture is informative of the number of contributors to the mixture. The package provides methods for computing the probability distribution of the number of distinct alleles in a mixture for a given set of allele frequencies. The mixture contributors may be related according to a provided pedigree.

License GPL (>= 2)

Encoding UTF-8

Imports Rcpp (>= 1.0.7), pedtools, ribd, partitions, methods

LinkingTo Rcpp

RoxygenNote 7.1.2

Suggests rmarkdown, knitr, testthat (>= 3.0.0), ggplot2

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation yes

Author Maarten Kruijver [aut, cre] (<`https://orcid.org/0000-0002-6890-7632`>),
James Curran [aut] (<`https://orcid.org/0000-0003-3323-6733`>)

Maintainer Maarten Kruijver <`maarten.kruijver@esr.cri.nz`>

Repository CRAN

Date/Publication 2022-04-29 08:00:09 UTC

R topics documented:

`expected_number_of_distinct_alleles` *Compute the expectation of the number of distinct alleles observed in a DNA mixture for a locus*

Description

For a given number of *independent* alleles, compute the mean and variance of the number of *distinct* alleles observed in a DNA mixture.

Usage

```
expected_number_of_distinct_alleles(number_of_independent_alleles, f, fst = 0)
```

Arguments

`number_of_independent_alleles`	Integer. Number of independent alleles in the mixture.
`f`	Numeric vector with allele frequencies
`fst`	Numeric value for sub-population correction (also known as theta)

Details

Due to allele sharing between DNA mixture contributors, the number of *distinct* alleles observed in a mixture is often less than the number of independent alleles in the mixture. For example, if mixture comprises two unrelated contributors, there are four independent alleles. Some of these four independent alleles may be of the same allelic type so that at least one and at most four distinct alleles are observed.

This function computes the probability distribution of the number of *distinct* alleles observed when the mixtures comprises a given number of *independent* alleles. Optionally, a sub-population correction may be applied by setting `fst>0`.

Value

Expected value (numeric of length one)

Examples

```
f <- c(A = 0.1, B = 0.2, C = 0.7)

e <- expected_number_of_distinct_alleles(3, f)

# by hand calculation
p <- pr_number_of_distinct_alleles(3, f)
e_by_hand <- sum(as.numeric(names(p)) * p)

stopifnot(isTRUE(all.equal(e, e_by_hand)))
```

`mean.pf` *Mean method for objects of class pf*

Description

Mean method for objects of class pf

Usage

```
## S3 method for class 'pf'
mean(x, by_locus = FALSE, ...)
```

Arguments

`x`	Object of class `pf`: output of pr_total_number_of_distinct_alleles
`by_locus`	If `TRUE` then the results will be returned locus by locus
`...`	other arguments that may p

Value

either a vector of means, one for each locus, or the sum of the locus means. The means are the expected number of alleles observed at each locus

Examples

```
freqs <- read_allele_freqs(system.file("extdata","FBI_extended_Cauc.csv",
                           package = "numberofalleles"))
p <- pr_total_number_of_distinct_alleles(contributors = c("U1","U2"),
                                          freqs = freqs)
mean(p)
var(p)
```

`numberofalleles` *numberofalleles*

Description

Probabilities for observing a number of distinct alleles in a forensic DNA mixture

Author(s)

Maarten Kruijver

`plot.pf` *Plot method for objects of class pf*

Description

Plot method for objects of class pf

Usage

```
## S3 method for class 'pf'
plot(
  x,
  lines = TRUE,
  points = TRUE,
  line_col = "black",
  point_col = "black",
  xlab = "Number of alleles (n)",
  ylab = expression(Pr(N == n)),
  add = FALSE,
  ...
)
```

Arguments

`x` An object of class pf

`lines` if `TRUE` then lines will be drawn from 0 to the probability value for each x value.

`points` if `TRUE` then points will be plotted at the probability value for each x value.

`line_col` the colour of the lines drawn for each probability mass

`point_col` the colour of the points plotted for each probability mass

`xlab` a label for the x axis, defaults to `"Number of alleles (n)"`

`ylab` a label for the y axis, defaults to `expression(Pr(N == n))`

`add` If `TRUE` then the plotting information will be added to the existing plot.

`...` Any other arguments that should be sent to `plot`, `arrows`, or `points`.

Value

No return value. Has the side effect of plotting to the active graphics device.

Examples

```
## Load allele frequencies
freqs <- read_allele_freqs(system.file("extdata","FBI_extended_Cauc.csv",
                            package = "numberofalleles"))

gf_loci <- c("D3S1358", "vWA", "D16S539", "CSF1PO", "TPOX", "D8S1179",
            "D21S11",  "D18S51", "D2S441", "D19S433", "TH01", "FGA",
            "D22S1045", "D5S818", "D13S317", "D7S820", "SE33",
            "D10S1248", "D1S1656", "D12S391", "D2S1338")

gf_freqs <- freqs[gf_loci]

## Compute the pedigrees for two and three siblings respectively
pedSibs2 = pedtools::nuclearPed(father = "F", mother = "M",
                                children = c("S1", "S2"))
pedSibs3 = pedtools::addChildren(father = "F", mother = "M",
                                 pedSibs2, ids = "S3")

## Compute the probability functions for each of 2 Unknowns, and 2 Sibs
p2U = pr_total_number_of_distinct_alleles(contributors = c("U1", "U2"),
                                          freqs = gf_freqs)
p2S =  pr_total_number_of_distinct_alleles(contributors = c("S1", "S2"),
                                           freqs = gf_freqs,
                                           pedigree = pedSibs2)

## And plot the two probability distribution functions on top of each other
plot(p2U,
     line_col = "red",
     point_col = "red",
     pch = 18,
     lwd = 2,
     ylim = c(0, 0.15),
     xlab = "Total Number of Alleles (TAC)",
     ylab = expression(Pr(N == n~"|"~ P)))

plot(p2S,
     add = TRUE,
     point_col = "blue",
     line_col = "blue",
     lwd = 2)

legend("topleft", legend = c("2 U", "2 S"), fill = c("red", "blue"), bty = "n")

## Compute the LR for the number of peaks given the
#  number of contributors and the pedigrees
lr = p2U$pf / p2S$pf
data.df = data.frame(log10lr = log10(lr), noa = p2U$noa)
```

```
## Plot the LR and a grid so that it's easy to see where
#  the LR becomes greater than 1
plot(log10lr~noa,
     data = data.df,
     axes = FALSE,
     xlab = "Total number of alleles, n",
     ylab = expression(log[10](LR(P[1],P[2]~"|"~N==n))),
     xaxs = "i", yaxs = "i",
     xlim = c(22,93),
     pch = 18)
axis(1)
y.exponents = seq(-20, 15, by = 5)
for(i in 1:length(y.exponents)){
  if(y.exponents[i] == 0)
    axis(2, at=y.exponents[i], labels="1", las = 1)
  else if(y.exponents[i] == 1)
    axis(2, at=y.exponents[i], labels="10", las = 1)
  else
    axis(2, at = y.exponents[i],
         labels = eval(substitute(
           expression(10^y),
           list(y = y.exponents[i]))),
         las = 1)
}
grid()
box()

## Let's look at 2 sibs versus 3 sibs

p3S = pr_total_number_of_distinct_alleles(contributors = c("S1", "S2", "S3"),
                                          freqs = gf_freqs,
                                          pedigree = pedSibs3)
plot(p3S,
     line_col = "green",
     point_col = "green",
     pch = 18,
     lwd = 2,
     ylim = c(0, 0.15),
     xlab = "Total Number of Alleles (TAC)",
     ylab = expression(Pr(N == n~"|"~ P)))

plot(p2S,
     add = TRUE,
     pch = 18,
     point_col = "blue",
     line_col = "blue",
     lwd = 2)
legend("topleft", legend = c("3 S", "2 S"), fill = c("green", "blue"), bty = "n")

## And finally two sibs and one unknown versus three sibs
p2SU = pr_total_number_of_distinct_alleles(contributors = c("S1", "S2", "U1"),
                                           freqs = gf_freqs,
                                              pedigree = pedSibs2)
plot(p3S,
     line_col = "green",
     point_col = "green",
     pch = 18,
     lwd = 2,
     ylim = c(0, 0.15),
     xlab = "Total Number of Alleles (TAC)",
     ylab = expression(Pr(N == n~"|"~ P)))
plot(p2SU,
     add = TRUE,
     pch = 18,
     point_col = "orange",
     line_col = "orange",
     lwd = 2)
legend("topleft", legend = c("3 S", "2 S + U"),
       fill = c("green", "orange"), bty = "n")
```

pr_independent_alleles

Compute the probability distribution of the number of independent alleles in a mixture with dropout

Description

Without dropout, each mixture contributor has two *independent* but not necessarily *distinct* alleles that are represented in the DNA mixture. If the probability of dropout for a mixture contributor is greater than zero, then the mixture contributor has either 0 (full dropout), 1 (partial dropout) or 2 (no dropout) independent alleles that are represented in the mixture. This function computes the probability distribution of the number of independent alleles that unrelated mixture contributors have in total for a locus given their dropout parameters.

Usage

```
pr_independent_alleles(dropout_prs)
```

Arguments

dropout_prs	Numeric vector. Dropout probabilities per contributor.

Value

A named numeric vector describing the probability distribution. Numeric values are the probabilities corresponding to the names describing integer values.

See Also

pr_independent_alleles_ped

Examples

```
# a dropout pr. of 0.5
p <- pr_independent_alleles(0.5)
stopifnot(all.equal(as.vector(p),
                    c(0.5^2, 2 * 0.5 * 0.5, (1-0.5)^2)))

# one contrib. without dropout and one with d=0.5
p1 <- pr_independent_alleles(c(0, 0.5))
stopifnot(identical(as.integer(names(p1)),
                    as.integer(names(p)) + 2L))
```

pr_independent_alleles_ped

Compute the probability distribution of the number of independent alleles in a mixture with dropout and related contributors

Description

When mixture contributors are related according to a pedigree, they may share some alleles identical by descent so that their total number of *independent* alleles is smaller than two times the number of contributors. The number of *independent* alleles can be further reduced if dropout plays a role. This function computes the probability distribution of the number of independent alleles that related mixture contributors have in total for a locus given their dropout parameters. Note that the number of *distinct* alleles that is observed at the locus is typically smaller than the number of independent alleles due to allele sharing.

Usage

```
pr_independent_alleles_ped(pedigree, ped_contributors, dropout_prs)
```

Arguments

`pedigree` ped object

`ped_contributors`
Character vector with unique names of contributors. Valid names are the names of pedigree members.

`dropout_prs` Numeric vector. Dropout probabilities per contributor.

Value

A named numeric vector describing the probability distribution. Numeric values are the probabilities corresponding to the names describing integer values.

See Also

pr_independent_alleles

Examples

```
# without dropout, a father-mother-child mixture has 4 indep. alleles
p <- pr_independent_alleles_ped(pedtools::nuclearPed(),
                                ped_contributors = as.character(1:3),
                                dropout_prs = rep(0, 3))
stopifnot(identical(p,
                    stats::setNames(1., "4")))
```

---

pr_number_of_distinct_alleles

Compute the probability distribution of the number of distinct alleles observed in a DNA mixture for a locus

---

Description

For a given number of *independent* alleles, compute the probability distribution of the number of *distinct* alleles observed in a DNA mixture.

Usage

```
pr_number_of_distinct_alleles(
  number_of_independent_alleles,
  f,
  fst = 0,
  brute_force = FALSE
)
```

Arguments

`number_of_independent_alleles`	Integer. Number of independent alleles in the mixture.
`f`	Numeric vector with allele frequencies
`fst`	Numeric value for sub-population correction (also known as theta)
`brute_force`	Logical. Should a brute force algorithm be used?

Details

Due to allele sharing between DNA mixture contributors, the number of *distinct* alleles observed in a mixture is often less than the number of independent alleles in the mixture. For example, if mixture comprises two unrelated contributors, there are four independent alleles. Some of these four independent alleles may be of the same allelic type so that at least one and at most four distinct alleles are observed.

This function computes the probability distribution of the number of *distinct* alleles observed when the mixtures comprises a given number of *independent* alleles. Optionally, a sub-population correction may be applied by setting `fst>0`.

An efficient way of computing the probability distribution was given by Tvedebrink (2014) and was slightly adapted by Kruijver & Curran (2022) to handle the case of an odd number of independent alleles. A much slower brute force algorithm is also implemented (argument `brute_force=TRUE`) for testing purposes.

Value

A named numeric vector describing the probability distribution. Numeric values are the probabilities corresponding to the names describing integer values.

References

M. Kruijver & J.Curran (2022). 'The number of alleles in DNA mixtures with related contributors', manuscript submitted

T. Tvedebrink (2014). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine; 128(3):427–37. doi: 10.1007/s0041401309513

Examples

```
f <- c(A = 0.1, B = 0.2, C = 0.7)

p <- pr_number_of_distinct_alleles(3, f)
p_by_hand <- c(sum(f^3), 1 - sum(f^3) - 6 * prod(f), 6 * prod(f))
stopifnot(all.equal(as.vector(p), p_by_hand))
```

`pr_total_number_of_distinct_alleles` *Compute the probability distribution of total number of distinct alleles in a DNA mixture*

Description

Compute the probability distribution of total number of distinct alleles in a DNA mixture

Usage

```
pr_total_number_of_distinct_alleles(
  contributors,
  freqs,
  pedigree,
  dropout_prs = rep(0, length(contributors)),
  fst = 0,
  loci = names(freqs)
)
```

Arguments

`contributors` Character vector with unique names of contributors. Valid names are "U1", "U2", ... for unrelated contributors or the names of pedigree members for related contributors.

`freqs` Allele frequencies (see read_allele_freqs)

`pedigree` (optionally) ped object

`dropout_prs` Numeric vector. Dropout probabilities per contributor. Defaults to zeroes.

`fst` Numeric. Defaults to 0.

`loci` Character vector of locus names (defaults to names attr. of `freqs`)

Details

A DNA mixture of n contributors contains $2n$ *independent* alleles per locus if the contributors are unrelated; fewer if they are related. This function computes the probability distribution of the total number of *distinct* alleles observed across all loci. Mixture contributors may be related according to an optionally specified pedigree. Optionally, a sub-population correction may be applied by setting `fst>0`.

The case where all contributors are unrelated was discussed by Tvedebrink (2014) and is implemented in the `DNAtools` package. Kruijver & Curran (2022) extended this to include related contributors by exploiting the multiPersonIBD function in the `ribd` package.

Value

an object of class `pf`. This is a list containing:

- pf. A named numeric vector describing the probability distribution of the total number of alleles. Numeric values are the probabilities corresponding to the names describing integer values.
- by_locus. A list of probability distributions by locus.
- noa. For convenience, an integer vector with the number of alleles corresponding to the probability distribution pf (the names attribute as integer vector)
- min. For convenience, the minimum of `noa`
- max. For convenience, the maximum of `noa`

References

M. Kruijver & J.Curran (2022). 'The number of alleles in DNA mixtures with related contributors', manuscript submitted

T. Tvedebrink (2014). 'On the exact distribution of the number of alleles in DNA mixtures', International Journal of Legal Medicine; 128(3):427–37. doi: 10.1007/s0041401309513

Examples

```
# define a pedigree of siblings S1 and S2 (and their parents)
ped_sibs <- pedtools::nuclearPed(children = c("S1", "S2"))

# define allele frequencies
freqs <- list(locus1 = c(0.1, 0.9),
              locus2 = c(0.25, 0.5, 0.25))

# compute dist. of number of alleles for two siblings and one unrelated persons
pr_total_number_of_distinct_alleles(contributors = c("S1","S2","U1"), freqs,
                                    pedigree = ped_sibs)

## GlobalFiler example (2 unrelated contributors)
freqs <- read_allele_freqs(system.file("extdata","FBI_extended_Cauc.csv",
package = "numberofalleles"))

gf_loci <- c("D3S1358", "vWA", "D16S539", "CSF1PO", "TPOX", "D8S1179",
             "D21S11",  "D18S51", "D2S441", "D19S433", "TH01", "FGA",
             "D22S1045", "D5S818", "D13S317", "D7S820", "SE33",
             "D10S1248", "D1S1656", "D12S391", "D2S1338")

p_gf <- pr_total_number_of_distinct_alleles(contributors = c("U1", "U2"),
                                            freqs = freqs, loci = gf_loci)

barplot(p_gf$pf)
```

---

read_allele_freqs   *Read allele frequencies in FSIgen format (.csv)*

Description

Read allele frequencies in FSIgen format (.csv)

Usage

```
read_allele_freqs(filename, remove_zeroes = TRUE, normalise = TRUE)
```

Arguments

filename	Path to csv file.
remove_zeroes	Logical. Should frequencies of 0 be removed from the return value? Default is TRUE.
normalise	Logical. Should frequencies be normalised to sum to 1? Default is TRUE.

Details

Reads allele frequencies from a .csv file. The file should be in FSIgen format, i.e. comma separated with the first column specifying the allele labels and one column per locus. The last row should be the number of observations. No error checking is done since the file format is only loosely defined, e.g. we do not restrict the first column name or the last row name.

Value

list

Examples

```
# below we read an allele freqs file that comes with the package
filename <- system.file("extdata","FBI_extended_Cauc.csv",package = "numberofalleles")
freqs <- read_allele_freqs(filename)
freqs # the output is just a list with an N attribute
```

---

var *Variance method for objects of class pf*

Description

Variance method for objects of class pf

Variance method for pf object

Usage

```
var(x, ...)

## S3 method for class 'pf'
var(x, by_locus = FALSE, ...)
```

Arguments

x	Object of class pf: output of pr_total_number_of_distinct_alleles
...	other arguments that may p
by_locus	If TRUE then the results will be returned locus by locus

Value

Either a vector of variances, one for each locus, or the sum of the locus variances. The variances are the variances of the number of alleles observed at each locus.

See Also

mean.pf

Examples

```
freqs <- read_allele_freqs(system.file("extdata","FBI_extended_Cauc.csv",
                           package = "numberofalleles"))
p <- pr_total_number_of_distinct_alleles(contributors = c("U1","U2"),
                                         freqs = freqs)
mean(p)
var(p)
```

---

variance_number_of_distinct_alleles

Compute the variance of the number of distinct alleles observed in a DNA mixture for a locus

---

Description

For a given number of *independent* alleles, compute the mean and variance of the number of *distinct* alleles observed in a DNA mixture.

Usage

```
variance_number_of_distinct_alleles(number_of_independent_alleles, f, fst = 0)
```

Arguments

number_of_independent_alleles	Integer. Number of independent alleles in the mixture.
f	Numeric vector with allele frequencies
fst	Numeric value for sub-population correction (also known as theta)

Details

Due to allele sharing between DNA mixture contributors, the number of *distinct* alleles observed in a mixture is often less than the number of independent alleles in the mixture. For example, if mixture comprises two unrelated contributors, there are four independent alleles. Some of these four independent alleles may be of the same allelic type so that at least one and at most four distinct alleles are observed.

This function computes the probability distribution of the number of *distinct* alleles observed when the mixtures comprises a given number of *independent* alleles. Optionally, a sub-population correction may be applied by setting `fst>0`.

Value

Variance (numeric of length one)

Examples

```
f <- c(A = 0.1, B = 0.2, C = 0.7)
v <- variance_number_of_distinct_alleles(3, f)

# now compute variance by hand from the full pr. dist of N
p <- pr_number_of_distinct_alleles(3, f)

n <- as.numeric(names(p))
p_n <- as.vector(p)

# compute expected value
ev <- expected_number_of_distinct_alleles(3, f)

# and variance by hand
v2 <- sum(p_n * (n - ev)^2)

stopifnot(all.equal(v,v2))
```

Index