Package 'dcifer'

August 10, 2022

Type Package

Title Genetic Relatedness Between Polyclonal Infections

Version 1.1.1

Maintainer Inna Gerlovina <innager@berkeley.edu>

Description An implementation of Dcifer (Distance for complex infections: fast estimation of relatedness), an identity by descent (IBD) based method to calculate genetic relatedness between polyclonal infections from biallelic and multiallelic data. The package includes functions that format and preprocess the data, implement the method, and visualize the results.
Gerlovina et al. (2022) <doi:10.1101/2022.04.14.488406>.

Imports stats, utils, graphics

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

URL https://github.com/EPPIcenter/dcifer,
https://eppicenter.github.io/dcifer/

Suggests knitr, rmarkdown

VignetteBuilder knitr

Depends R (>= 3.5.0)

NeedsCompilation yes

Author Inna Gerlovina [aut, cre] (<https://orcid.org/0000-0002-7772-7473>)

Repository CRAN

Date/Publication 2022-08-10 15:50:14 UTC

R topics documented:

`calcAfreq` *Calculate Allele Frequencies*

Description

Calculates population allele frequencies from data, adjusting for COI.

Usage

```
calcAfreq(dsmp, coi, tol = 1e-04, qstart = 0.5)
```

Arguments

`dsmp`	a list with each element corresponding to one sample.
`coi`	a vector containing complexity of infection for each sample.
`tol`	convergence tolerance for frequency estimates.
`qstart`	a starting value for frequencies.

Value

A list of allele frequencies, where each element is a numeric vector containing frequencies for a single locus.

Examples

```
coi   <- getCOI(dsmp, lrank = 2)          # estimate COI first
afreq <- calcAfreq(dsmp, coi, tol = 1e-5)
```

dres *Dcifer results*

Description

Results of relatedness estimation.

Usage

```
dres
```

Format

A three-dimensional array with 52 columns, 52 rows, and 4 matrices. Dimension names correspond to sample ID's (rows and columns) and types of results `c("estimate", "p_value", "CI_lower", "CI_upper")` (matrices).

dsmp *Sample data*

Description

Microhaplotype data from two clinics in Mozambique. Samples are sorted by location.

Usage

```
dsmp
```

Format

A list of 52 elements (samples), each element is a list of 87 elements (loci), which are integer vectors (alleles).

generateReval *Grid of Parameter Values*

Description

Generates a grid of parameter values to evaluate over.

Usage

```
generateReval(M, rval = NULL, nr = NULL)
```

Arguments

M	an integer.
rval	a numeric vector with parameter values for the grid. If not provided, it will be generated by dividing $[0, 1]$ into nr equal intervals.
nr	an integer. Ignored if rval is provided.

Value

A a matrix with M rows and nr + 1 or length(rval) columns.

Examples

```
reval <- generateReval(M = 2, nr = 1e2)
dim(reval)
```

getCOI *Calculate COI*

Description

Calculates complexity of infection for a list of samples, using the number of detected alleles.

Usage

```
getCOI(dsmp, lrank = 2)
```

Arguments

dsmp	a list with each element corresponding to one sample.
lrank	the rank of the locus that will determine a sample's COI (loci are ranked by the number of detected alleles).

Value

a vector with estimated COI for each sample.

Examples

```
coi <- getCOI(dsmp, lrank = 2)
```

---

ibdDat *Pairwise Relatedness*

Description

Provides pairwise relatedness estimates within a dataset or between two datasets along with optional p-values and confidence intervals (CI).

Usage

```
ibdDat(
  dsmp,
  coi,
  afreq,
  dsmp2 = NULL,
  coi2 = NULL,
  pval = TRUE,
  confint = FALSE,
  rnull = 0,
  alpha = 0.05,
  nr = 1000,
  reval = NULL
)
```

Arguments

dsmp	a list with each element corresponding to one sample.
coi	a vector containing complexity of infection for each sample.
afreq	a list of allele frequencies. Each element of the list corresponds to a locus.
dsmp2	a list representing a second dataset.
coi2	a vector with complexities of infection for a second dataset.
pval	a logical value specifying if p-values should be returned.
confint	a logical value specifying if confidence intervals should be returned.
rnull	a null value of relatedness parameter for hypothesis testing (needed if pval = TRUE).
alpha	significance level for a $1 - \alpha$ confidence region.

nr	an integer specifying precision of the estimate: resolution of a grid of parameter values ($[0, 1]$ divided into nr equal intervals), over which the likelihood will be calculated. Ignored if non-null reval is provided.
reval	a vector or a single-row matrix. A grid of parameter values, over which the likelihood will be calculated.

Details

For this function, M is set to 1. If confint = FALSE, Newton's method is used to find the estimates, otherwise the likelihood is calculated for a grid of parameter values.

Value

A matrix if pval and confint are FALSE and 3-dimensional arrays otherwise. The matrices are lower triangular if distances are calculated within a dataset. For a 3-dimensional array, stacked matrices contain relatedness estimates, p-values, and endpoints of confidence intervals (if requested).

See Also

ibdPair for genetic relatedness between two samples and ibdEstM for estimating the number of related pairs of strains.

Examples

```
coi   <- getCOI(dsmp, lrank = 2)           # estimate COI
afreq <- calcAfreq(dsmp, coi, tol = 1e-5)  # estimate allele frequencies

# subset of samples for faster processing
i1 <- 1:15     # from Maputo
i2 <- 31:40    # from Inhambane
isub <- c(i1, i2)

# matrix is returned
dres1 <- ibdDat(dsmp[isub], coi[isub], afreq, pval = FALSE)
dim(dres1)

# test a null hypothesis H0: r = 0, change precision
dres2 <- ibdDat(dsmp[isub], coi[isub], afreq, pval = TRUE, rnull = 0,
                nr = 1e2)
dim(dres2)

# test H0: r = 0.2, include 99% confidence intervals
dres3 <- ibdDat(dsmp[isub], coi[isub], afreq, pval = TRUE, confint = TRUE,
                rnull = 0.2, alpha = 0.01)
dres3[2, 1, ]

# pairwise relatedness between two datasets, H0: r = 0
drbetween <- ibdDat(dsmp[i1], coi[i1], afreq,
                    dsmp2 = dsmp[i2], coi2 = coi[i2])
dim(drbetween)
drbetween[1, 2, ]
```

```
sum(is.na(drbetween[, , 1]))
```

---

ibdEstM *Estimate Relatedness and a Number of Related Strains*

Description

Estimates the number of related pairs of strains between two infections along with corresponding relatedness estimates and optional inference.

Usage

```
ibdEstM(
  pair,
  coi,
  afreq,
  Mmax = 6,
  pval = FALSE,
  confreg = FALSE,
  llik = FALSE,
  rnull = 0,
  alpha = 0.05,
  equalr = FALSE,
  freqlog = FALSE,
  nrs = c(1000, 100, 32, 16, 12, 10),
  revals = NULL,
  tol0 = 1e-09,
  logrs = NULL,
  nevals = NULL,
  nloc = NULL
)
```

Arguments

`pair`	a list of length two containing data for a pair of samples.
`coi`	a vector containing complexity of infection for each sample.
`afreq`	a list of allele frequencies. Each element of the list corresponds to a locus.
`Mmax`	a maximum number of related pairs of strains to evaluate over. If greater than `min(coi)`, will be set to `min(coi)`.
`pval, confreg, llik`	logical values specifying if p-value, confidence region, and log-likelihood for a range of r values should be returned.
`rnull`	a null value of relatedness parameter for hypothesis testing (needed if `pval = TRUE`).
`alpha`	significance level for a $1 - \alpha$ confidence region.

equalr	a logical value. If TRUE, the same level of relatedness is assumed for M pairs of strains (r_1 = ... = r_M).
freqlog	a logical value indicating if afreq is on the log scale.
nrs	an integer vector where i'th element correspons to $M = i$ and indicates precision of the estimate (resolution of a grid of parameter values). Ignored if non-null revals is provided.
revals	a list where i'th element corresponds to $M = i$ and is a matrix representing a grid of parameter values (a matrix where each column represents a single $(r_1, ..., r_M)$ combination).
tol0	a tolerance value for an estimate to be considered zero.
logrs	a list where i'th element corresponds to $M = i$ and is a list as returned by logReval.
nevals	a vector where i'th element corresponds to $M = i$ and provides the number of relatedness values/combinations to evaluate over.
nloc	the number of loci.

Value

A named list if multiple output logical values are TRUE - or a vector if only rhat = TRUE. The output includes:

- a relatedness estimate (numeric vector of length corresponding to the estimated number of related pairs);
- a p-value if pval = TRUE;
- parameter values from the grid in revals that are within the confidence region if confreg = TRUE;
- log-likelihood values for the parameter grid in revals if llik = TRUE.

See Also

ibdPair for estimates of relatedness between two samples and ibdDat for pairwise relatedness estimates within a dataset or between two datasets.

Examples

```
coi   <- getCOI(dsmp, lrank = 2)           # estimate COI
afreq <- calcAfreq(dsmp, coi, tol = 1e-5)  # estimate allele frequencies

# two samples
ipair <- c(21, 17)
# for higher COI: c(33, 5): COI = 5-6; c(37, 20): 4-3, c(41, 50): 5-4

Mmax  <- min(coi[ipair])
# choose resolution of the grid for different M
nrs   <- c(1e3, 1e2, 32, 16, 12, 10)[1:Mmax]
revals <- mapply(generateReval, 1:Mmax, nr = nrs)
```

```
(res1 <- ibdEstM(dsmp[ipair], coi[ipair], afreq, Mmax = Mmax, equalr = FALSE,
                 reval = revals))
(res2 <- ibdEstM(dsmp[ipair], coi[ipair], afreq, Mmax = Mmax, equalr = TRUE))
# number of related pairs of strains (M')
sum(res1 > 0)
sum(res2 > 0)  # can be 0's
```

---

ibdPair *Relatedness Between Two Samples*

Description

Provides estimates of relatedness between a pair of samples along with an optional support curve and inference.

Usage

```
ibdPair(
  pair,
  coi,
  afreq,
  M,
  rhat = TRUE,
  pval = FALSE,
  confreg = FALSE,
  llik = FALSE,
  maxllik = FALSE,
  rnull = 0,
  alpha = 0.05,
  equalr = FALSE,
  mnewton = NULL,
  freqlog = FALSE,
  nr = 1000,
  reval = NULL,
  tol = NULL,
  logr = NULL,
  neval = NULL,
  inull = NULL,
  nloc = NULL
)
```

Arguments

pair	a list of length two containing data for a pair of samples.
coi	a vector containing complexity of infection for each sample.
afreq	a list of allele frequencies. Each element of the list corresponds to a locus.

M	the number of related pairs of strains.
rhat, pval, confreg, llik, maxllik	logical values specifying if relatedness estimate, p-value, confidence region, log-likelihood for a range of r values, and maximum log-likelihood should be returned.
rnull	a null value of relatedness parameter for hypothesis testing (needed if pval = TRUE).
alpha	significance level for a $1-\alpha$ confidence region.
equalr	a logical value. If TRUE, the same level of relatedness is assumed for M pairs of strains (r_1 = ... = r_M).
mnewton	a logical value. If TRUE, Newton's method, adapted for a bounded parameter space, will be used to find MLE. If mnewton is not specified, it will be set to TRUE if M = 1, confreg = FALSE, and llik = FALSE.
freqlog	a logical value indicating if afreq is on the log scale.
nr	an integer specifying precision of the estimate: resolution of a grid of parameter values ($[0,1]$ divided into nr equal intervals), over which the likelihood will be calculated. Ignored if non-null reval is provided.
reval	a matrix representing a grid of $(r_1, ..., r_M)$ combinations, over which the likelihood will be calculated. Each column is a single combination.
tol	tolerance for calculating an estimate if mnewton = TRUE. Set to 1/nr if not provided.
logr	a list as returned by logReval with logs of reval and other quantities.
neval	the number of relatedness values/combinations to evaluate over.
inull	an index of the value/column of reval that is closest to rnull.
nloc	the number of loci.

Details

Handling of irregular cases:

- Allele with population frequency of 0 is present: locus is skipped (does not contribute any information).
- Number of unique alleles at a locus is greater than COI: COI will be increased for that locus only.

Value

A named list if multiple output logical values are TRUE - or a vector if only rhat = TRUE or llik = TRUE. Depending on these logical values, the following quantities are included:

- If rhat = TRUE, a relatedness estimate (a vector of length 1 if equalr = TRUE or of length M if equalr = FALSE);
- If pval = TRUE, a p-value;
- If confreg = TRUE, relatedness parameter values from the grid reval that are within $1-\alpha$ confidence region;

- If `llik = TRUE`, log-likelihood values for relatedness parameter grid (provided in `reval` or determined by `nr`);
- If `maxllik = TRUE`, maximum log-likelihood.

See Also

`ibdEstM` for estimating the number of related pairs of strains and `ibdDat` for processing multi-sample data.

Examples

```
coi   <- getCOI(dsmp, lrank = 2)
afreq <- calcAfreq(dsmp, coi, tol = 1e-5)

# two samples
ipair <- c(21, 17)
pair <- dsmp[ipair]
coip <-  coi[ipair]
M    <- 2

res1 <- ibdPair(pair, coip, afreq, M = M, confreg = TRUE, alpha = 0.05,
                equalr = FALSE, reval = revals[[M]])
res2 <- ibdPair(pair, coip, afreq, M = M, llik = TRUE,
                equalr = TRUE, reval = revals[[1]])
res1$rhat
rep(res2$rhat, M)

# plot confidence region
creg <- cbind(res1$confreg, res1$confreg[2:1, ])
plot(creg[1, ], creg[2, ], xlim = c(0, 1), ylim = c(0, 1), pch = 15,
     cex = 0.6, col = "cadetblue3", xlab = expression(hat(r)[1]),
     ylab = expression(hat(r)[2]))
points(res1$rhat, rev(res1$rhat), pch = 16)

# plot log-likelihood
plot(revals[[1]], res2$llik, type = "l", xlab = "r", ylab = "log-likelihood")

ipair <- c(41, 50)
pair <- dsmp[ipair]
coip <-  coi[ipair]

# rtotal at different values of M with and without equality constraint
Mmax <- min(coip)
for (M in 1:Mmax) {
  print(paste0("M = ", M))
  print(c(sum(ibdPair(pair, coip, afreq, M = M, pval = FALSE,
                      equalr = FALSE, reval = revals[[M]])),
          ibdPair(pair, coip, afreq, M = M, pval = FALSE, equalr = TRUE)*M))
  cat("\n")
}

# M = 1
```

```
# log-likelihood for specific r values
ibdPair(pair, coip, afreq, M = 1, rhat = FALSE, pval = FALSE, llik = TRUE,
        reval = c(0, 0.15, 0.38, 1))
# grid vs Newton's method
system.time(
  ibdPair(pair, coip, afreq, M = 1, mnewton = TRUE,  tol = 1e-5))
system.time(
  ibdPair(pair, coip, afreq, M = 1, mnewton = FALSE, nr  = 1e5))
```

---

logReval *Logarithms of* reval

Description

Calculates logarithms of reval and 1 - reval, as well as other associated quantities.

Usage

```
logReval(reval, M = NULL, neval = NULL, equalr = FALSE)
```

Arguments

reval	a matrix representing a grid of $(r_1, ..., r_M)$ combinations, over which the likelihood will be calculated. Each column is a single combination.
M	the number of related pairs of strains.
neval	the number of relatedness values/combinations to evaluate over.
equalr	a logical value. If TRUE, the same level of relatedness is assumed for M pairs of strains (r_1 = ... = r_M).

Details

For equalr = TRUE relatedness estimation, reval should be a 1 x neval matrix.

Value

A list of length 5 that contains log(reval), log(1 - reval), the number of reval = 1 for each column, the number of 0 < reval < 1 for each column, and sum(log(1 - reval[reval < 1])) for each column.

Examples

```
reval <- generateReval(M = 2, nr = 1e2)
logr  <- logReval(reval, M = 2, equalr = FALSE)

reval <- generateReval(M = 1, nr = 1e3)
logr3  <- logReval(reval, M = 3, equalr = TRUE)
logr1  <- logReval(reval, M = 1)
all(logr3$sum1r == logr1$sum1r*3)
```

`matchAfreq` *Match Samples and Allele Frequencies*

Description

Checks if the list containing sample data is conformable to provided population allele frequencies and reformats it if needed.

Usage

```
matchAfreq(dsmp, afreq)
```

Arguments

`dsmp`	a list with each element corresponding to one sample.
`afreq`	a list of allele frequencies. Each element of the list corresponds to a locus.

Details

The function reorders loci and alleles in `dsmp` to match those in `afreq` and inserts alleles into `dsmp` if they are present in `afreq` and not in `dsmp`; doesn't handle cases when alleles are present in `dsmp` but not in `afreq`. Allele names are required for this procedure.

Value

A list of the same length as `dsmp`, with each element matching the lengths and the names of `afreq` and its elements.

See Also

`readDat` and `readAfreq` for reading in and reformating data.

Examples

```
afile  <- system.file("extdata", "MozAfreq.csv", package = "dcifer")
afreq2 <- readAfreq(afile, lvar = "locus", avar = "allele", fvar = "freq")
dsmp2  <- matchAfreq(dsmp, afreq2)
```

plotColorbar *Colorbar*

Description

Creates a colorbar for a plot.

Usage

```
plotColorbar(
  rlim = c(0, 1),
  by = 0.1,
  at = NULL,
  horiz = FALSE,
  col = grDevices::hcl.colors(301, "YlGnBu", rev = TRUE),
  ...
)
```

Arguments

rlim	the range of values to be represented by colors.
by	increment size for tickmark locations. Ignored if at is provided.
at	a vector of tickmark locations.
horiz	a logical value specifying if the colorbar should be drawn horizontally.
col	the colors for the colorbar.
...	other graphical parameters.

Details

The colorbar will fill the whole plotting region, which needs to be specified outside of this function to control proportions and location of the colorbar (see examples). To match the colors in the main plot, rlim values should be the same for plotRel and plotColorbar; if rlim = NULL or rlim = NA in plotRel, provide the actual range of relatedness estimates for plotColorbar (see examples).

Value

NULL; called for plotting.

See Also

plotRel for plotting relatedness estimates.

Examples

```
parstart <- par(no.readonly = TRUE)   # save starting graphical parameters

# colorbar on the side of the main plot
layout(matrix(1:2, 1), width = c(7, 1))
par(mar = c(2, 0, 2, 0) + 0.1)
# make symmetric matrix
dmat <- dres[, , "estimate"]
dmat[upper.tri(dmat)] <- t(dmat)[upper.tri(t(dmat))]
isig <- which(dres[, , "p_value"] <= 0.05, arr.ind = TRUE)
plotRel(dmat, draw_diag = TRUE, isig = rbind(isig, isig[, 2:1]))
abline(v = 26, h = 26, col = "gray45", lty = 5)
par(mar = c(2, 1, 2, 2) + 0.1)
plotColorbar()

# shorter colorbar, tick mark locations provided
par(mar = c(2, 0, 2, 0) + 0.1)
plotRel(dmat, draw_diag = TRUE, isig = rbind(isig, isig[, 2:1]))
par(mar = c(5, 0.5, 5, 2.5) + 0.1)
plotColorbar(at = c(0.0625, 0.125, 0.25, 0.5, 0.78))
par(parstart)

# triangular matrix, inset horizontal colorbar
par(mar = c(1, 1, 1, 1))
plotRel(dres, rlim = NULL, draw_diag = TRUE, border_diag = 1, alpha = 0.05)
par(fig = c(0.3, 1.0, 0.73, 0.83), new = TRUE)
rlim <- range(dres[, , 1], na.rm = TRUE)
plotColorbar(rlim = rlim, at = c(0.2, 0.4, 0.6, 0.8), horiz = TRUE)
par(parstart)
```

---

plotRel *Plot Relatedness Estimates*

Description

Represents a matrix of pairwise relatedness estimates with colors corresponding to the levels of relatedness. Optionally, also outlines results of a hypothesis testing. The plot follows a matrix layout.

Usage

```
plotRel(
  r,
  rlim = c(0, 1),
  isig = NULL,
  alpha = NULL,
  col = grDevices::hcl.colors(101, "YlGnBu", rev = TRUE),
  draw_diag = FALSE,
```

```
  col_diag = "gray",
  border_diag = NA,
  lwd_diag = 0.5,
  border_sig = "orangered2",
  lwd_sig = 1.5,
  xlab = "",
  ylab = "",
  add = FALSE,
  idlab = FALSE,
  side_id = c(1, 2),
  col_id = 1,
  cex_id = 0.5,
  srt_id = NULL,
  ...
)
```

Arguments

r	a matrix or a 3-dimensional array as returned by ibdDat.
rlim	the range of values for colors. If NULL or NA, will be calculated from r.
isig	a matrix with two columns providing indices of relatedness matrix entries to be outlined ("significant" sample pairs). Takes precedence over alpha.
alpha	significance level for hypothesis testing; determines relatedness matrix entries to be outlined. Ignored if isig is not NULL.
col	the colors for the range of relatedness values.
draw_diag	a logical value specifying if diagonal cells should be distinguished from others by a separate color.
col_diag, border_diag, lwd_diag	the color for the fill, the color for the border, and the line width for the border of diagonal entries. Ignored if draw_diag = FALSE.
border_sig, lwd_sig	the color and the line width for outlining entries specified by isig or alpha.
xlab, ylab	axis labels.
add	a logical value specifying if the graphics should be added to the existing plot (useful for triangular matrices).
idlab	a logical value specifying if sample ID's should be displayed.
side_id	an integer vector specifying plot sides for sample ID labels.
col_id, cex_id	numeric vectors for the color and the size of sample ID labels.
srt_id	a vector of the same length as side_id specifying rotation angles for sample ID labels. If NULL, the labels will be perpendicular to the axes.
...	other graphical parameters.

Value

NULL; called for plotting.

See Also

plotColorbar for a colorbar.

Examples

```
parstart <- par(no.readonly = TRUE)   # save starting graphical parameters

par(mar = c(0.5, 0.5, 0.5, 0.5))
plotRel(dres, alpha = 0.05, draw_diag = TRUE)

# draw log of p-values in the upper triangle
pmat <- matrix(NA, nrow(dres), ncol(dres))
pmat[upper.tri(pmat)] <- t(log(dres[, , "p_value"]))[upper.tri(pmat)]
pmat[pmat == -Inf] <- min(pmat[is.finite(pmat)])
plotRel(pmat, rlim = NULL, draw_diag = TRUE, col = hcl.colors(101, "PuRd"),
        add = TRUE, col_diag = "slategray2", border_diag = 1)

# symmetric matrix, outline significant in upper triangle, display sample ID
par(mar = c(3, 3, 0.5, 0.5))
dmat <- dres[, , "estimate"]
dmat[upper.tri(dmat)] <- t(dmat)[upper.tri(t(dmat))]
isig <- which(dres[, , "p_value"] <= 0.05, arr.ind = TRUE)
col_id <- rep(c("plum4", "lightblue4"), each = 26)
plotRel(dmat, isig = isig[, 2:1], draw_diag = TRUE, idlab = TRUE,
        col_id = col_id)
abline(v = 26, h = 26, col = "gray45", lty = 5)

# rotated sample ID labels on all sides
par(mar = c(3, 3, 3, 3))
plotRel(dmat, isig = rbind(isig, isig[, 2:1]), border_sig = "magenta2",
        draw_diag = TRUE, idlab = TRUE, side_id = 1:4, col_id = col_id,
        srt_id = c(-55, 25, 65, -35))
par(parstart)
```

---

probUxUy *Likelihood for U_x, U_y*

Description

Calculates log-likelihood for a pair of samples at a single locus.

Usage

```
probUxUy(
  Ux,
  Uy,
  nx,
  ny,
```

```
  probs,
  M,
  logj,
  factj,
  equalr = FALSE,
  mnewton = TRUE,
  reval = NULL,
  logr = NULL,
  neval = NULL
)
```

Arguments

Ux, Uy	sets of unique alleles for two samples at a given locus. Vectors of indices corresponding to ordered probabilities in probs.
nx, ny	complexity of infection for two samples. Vectors of length 1.
probs	a vector of population allele frequencies (on a log scale) at a given locus. It is not checked if frequencies on a regular scale sum to 1.
M	the number of related pairs of strains.
logj, factj	numeric vectors containing precalculated logarithms and factorials.
equalr	a logical value. If TRUE, the same level of relatedness is assumed for M pairs of strains (r_1 = ... = r_M).
mnewton	a logical value. If TRUE, the coefficients for using Newton's method will be calculated.
reval	a matrix representing a grid of $(r_1, ..., r_M)$ combinations, over which the likelihood will be calculated. Each column is a single combination.
logr	a list of length 5 as returned by logReval.
neval	the number of relatedness values/combinations to evaluate over.

Value

- If mnewton = TRUE, a vector of length 2 containing coefficients for fast likelihood calculation;
- If mnewton = FALSE, a vector of length neval containing log-likelihoods for a range of parameter values.

Examples

```
Ux <- c(1, 3, 7)                      # detected alleles at locus t
Uy <- c(2, 7)
coi <- c(5, 6)
aft <- runif(7)                       # allele frequencies for locus t
aft <- log(aft/sum(aft))

logj  <- log(1:max(coi))
factj <- lgamma(0:max(coi) + 1)

# M = 2, equalr = FALSE
```

```
M <- 2
reval <- generateReval(M, nr = 1e2)
logr  <- logReval(reval, M = M)
llikt <- probUxUy(Ux, Uy, coi[1], coi[2], aft, M, logj, factj,
                  equalr = FALSE, logr = logr, neval = ncol(reval))
length(llikt)

# M = 2, equalr = TRUE
reval <- matrix(seq(0, 1, 0.001), 1)
logr  <- logReval(reval, M = M, equalr = TRUE)
llikt <- probUxUy(Ux, Uy, coi[1], coi[2], aft, M, logj, factj,
                  equalr = TRUE, logr = logr, neval = ncol(reval))

# M = 1, mnewton = FALSE
M <- 1
reval <- matrix(seq(0, 1, 0.001), 1)
logr  <- logReval(reval, M = M)
llikt <- probUxUy(Ux, Uy, coi[1], coi[2], aft, M, logj, factj,
                  mnewton = FALSE, reval = reval, logr = logr,
                  neval = ncol(reval))

# M = 1, mnewton = TRUE
probUxUy(Ux, Uy, coi[1], coi[2], aft, M, logj, factj, mnewton = TRUE)
```

`read` *Read and Reformat Data*

Description

Reads data from `csv` files and reformats these data for further processing. Original data are assumed to be in a long format, with one row per allele.

Usage

```
readDat(sfile, svar, lvar, avar, ...)

readAfreq(afile, lvar, avar, fvar, ...)
```

Arguments

`sfile`	the name of the file containing sample data.
`svar`	the name of the variable for sample ID.
`lvar`	the name of the variable for locus/marker.
`avar`	the name of the variable for allele/haplotype.
`...`	additional arguments for `read.csv()`.
`afile`	the name of the file containing population allele frequencies.
`fvar`	the name of the variable for population allele frequiency.

Value

For `readDat`, a list with elements corresponding to samples. Each element of the list is itself a list of binary vectors, one vector for each locus. For `readAfreq`, a list with elements corresponding to loci. The frequencies at each locus are normalized and sum to 1. Samples, loci, and alleles are ordered by their IDs/names.

See Also

`matchAfreq` for making sure that the list containing sample data is conformable to provided population allele frequencies.

Examples

```
sfile <- system.file("extdata", "MozParagon.csv", package = "dcifer")
dsmp  <- readDat(sfile, svar = "sampleID", lvar = "locus", avar = "allele")

afile  <- system.file("extdata", "MozAfreq.csv", package = "dcifer")
afreq2 <- readAfreq(afile, lvar = "locus", avar = "allele", fvar = "freq")
dsmp2  <- matchAfreq(dsmp, afreq2)
```

`revals` *Parameter grid*

Description

Precalculated parameter grids for a range of values of M (from 1 to 5).

Usage

```
revals
```

Format

A list of length 5, where each element corresponds to a single value of M and is a matrix with M rows. Each column of a matrix is a r_1 = ... = r_M) combination.

Index