Package 'diemr'

September 11, 2025

Title Diagnostic Index Expectation Maximisation in R

Version 1.5

Description Likelihood-based genome polarisation finds which alleles of genomic markers belong to which side of the barrier.
Co-estimates which individuals belong to either side of the barrier and barrier strength. Uses expectation maximisation in likelihood framework. The method is described in Baird et al. (2023) <doi:10.1111/2041-210X.14010>.

BugReports https://github.com/StuartJEBaird/diem/issues

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

Imports zoo, vcfR, data.table, circlize

VignetteBuilder knitr

NeedsCompilation no

Author Natalia Martinkova [aut, cre] (ORCID: <https://orcid.org/0000-0003-4556-4363>), Stuart Baird [aut] (ORCID: <https://orcid.org/0000-0002-7144-9919>)

Maintainer Natalia Martinkova <martinkova@ivb.cz>

Repository CRAN

Date/Publication 2025-09-11 11:30:11 UTC

Contents

`brenthis` *Dataset of Butterfly Genotypes*

Description

A subset of single nucleotide polymorphisms in butterflies of the genus *Brenthis*.

Format

vcf file with 13 individuals and 4 markers.

Details

The data is used to test conversion of genotype data from vcf to diem format with function `vcf2diem`.

Examples

```
filename <- system.file("extdata", "brenthis.vcf", package = "diemr")
```

`CheckDiemFormat` *diem input file checker*

Description

Checks format of files with genotype data.

Usage

```
CheckDiemFormat(files, ChosenInds, ploidy)
```

Arguments

files	A character vector with paths to files with genotypes.
ChosenInds	A numeric or logical vector of indices of individuals to be included in the analysis.
ploidy	A logical or a list of length equal to length of files. Each element of the list contains a numeric vector with ploidy numbers for all individuals specified in the files.

Details

The input file must have genotypes of one marker for all individuals on one line. The line must start with a letter "S" and contain only characters "_" or "U" for unknown genotypes or a third/fourth allele, "0" for homozygots for allele 1, "1" for heterozygots, and "2" for homozygots for allele 2. Check the vignette with `browseVignettes(package = "diemr")` for the example of the input format.

Ploidies must be given as a list with each element corresponding to a genomic compartment (aka a file). For each compartment, the numeric vector specifying ploidies of all individuals chosen for the specific analysis must be given.

Value

Returns invisible TRUE if all files are executable by diem. Exits with informative error messages otherwise, specifying file names and lines with potential problems. When too many lines contain problems, the first six are given.

Examples

```
# set up input genotypes file names, ploidies and selection of individual samples
inputFile <- system.file("extdata", "data7x3.txt", package = "diemr")
ploidies <- list(c(2, 1, 2, 2, 2, 1, 2))
inds <- 1:7

# check input data
CheckDiemFormat(files = inputFile, ploidy = ploidies, ChosenInds = inds)
#  File check passed: TRUE
#  Ploidy check passed: TRUE
```

---

diem *Diagnostic Index Expectation Maximisation*

Description

Estimates how to assign alleles in a genome to maximise the distinction between two unknown groups of individuals. Using expectation maximisation (EM) in likelihood framework, diem provides marker polarities for importing data, their likelihood-based diagnostic index and its support for all markers, and hybrid indices for all individuals.

Usage

```
diem(
  files,
  ploidy = FALSE,
  markerPolarity = FALSE,
  ChosenInds,
  ChosenSites = "all",
  epsilon = 0.99999,
  verbose = FALSE,
  nCores = parallel::detectCores() - 1,
  maxIterations = 50,
  ...
)
```

Arguments

files	A character vector with paths to files with genotypes.
ploidy	A logical or a list of length equal to length of files. Each element of the list contains a numeric vector with ploidy numbers for all individuals specified in the files.
markerPolarity	FALSE or a list of logical vectors.
ChosenInds	A numeric or logical vector of indices of individuals to be included in the analysis.
ChosenSites	A logical vector indicating which sites are to be included in the analysis.
epsilon	A numeric, specifying how much the hypothetical diagnostic markers should contribute to the likelihood calculations. Must be in [0,1), keeping tolerance setting of the R session in mind.
verbose	Logical or character with path to directory where run diagnostics will be saved.
nCores	A numeric number of cores to be used for parallelisation. Must be nCores = 1 on Windows.
maxIterations	A numeric.
...	additional arguments.

Details

Given two alleles of a marker, one allele can belong to one side of a barrier to geneflow and the other to the other side. Which allele belongs where is a non-trivial matter. A marker state in an individual can be encoded as 0 if the individual is homozygous for the first allele, and 2 if the individual is homozygous for the second allele. Marker polarity determines how the marker will be imported. Marker polarity equal to FALSE means that the marker will be imported as-is. A marker with polarity equal to TRUE will be imported with states 0 mapped as 2 and states 2 mapped as 0, in effect switching which allele belongs to which side of a barrier to geneflow.

When markerPolarity = FALSE, diem uses random null polarities to initiate the EM algorithm. To fix the null polarities, markerPolarity must be a list of length equal to the length of the files argument, where each element in the list is a logical vector of length equal to the number of markers (rows) in the specific file.

Ploidy needs to be given for each compartment and for each individual. For example, for a dataset of three diploid mammal males consisting of an autosomal compartment, an X chromosome compartment and a Y chromosome compartment, the ploidy list would be `ploidy = list(rep(2, 3), rep(1, 3), rep(1, 3))`. If the dataset consisted of one male and two females, ploidy for the sex chromosomes should be vectors reflecting that females have two X chromosomes, but males only one, and females have no Y chromosomes: `ploidy = list(rep(2, 3), c(1, 2, 2), c(1, 0, 0))`.

When a subset of individuals is used to inform the genome polarisation in the `ChosenInds` argument, `ploidy` must still be provided for all individuals included in the `files`.

`ChosenInds` should preferably be numeric values within the range from 1 to the number of individuals in the `files`. Logical vectors must have a length equal to the number of individuals in the `files`.

When `verbose = TRUE`, `diem` will output multiple files with information on the iterations of the EM algorithm, including tracking marker polarities and the respective likelihood-based diagnostics. See vignette `vignette("Understanding-genome-polarisation-output-files", package = "diemr")` for a detailed explanation of the individual output files.

Value

A list including suggested marker polarities, diagnostic indices and support for all markers, four genomic state counts matrix for all individuals, and polarity changes for the EM iterations.

Note

To ensure that the data input format of the genotype files, ploidies and individual selection are readable for `diem`, first use CheckDiemFormat. Fix all errors, and run `diem` only once the checks all passed.

The working directory or a folder optionally specified in the `verbose` argument must have write permissions. `diem` will store temporary files in the location and output results files.

The grain for parallelisation is the compartment `files`.

See Also

CheckDiemFormat

Examples

```
# set up input genotypes file names, ploidies and selection of individual samples
inputFile <- system.file("extdata", "data7x3.txt", package = "diemr")
ploidies <- list(c(2, 1, 2, 2, 2, 1, 2))
inds <- 1:6

# check input data
CheckDiemFormat(files = inputFile, ploidy = ploidies, ChosenInds = inds)
#  File check passed: TRUE
#  Ploidy check passed: TRUE

# run diem
## Not run:
# diem will write temporal files during EM iterations
```

```
# prior to running diem, set the working directory to a location with write permission
fit <- diem(files = inputFile, ChosenInds = inds, ploidy = ploidies, nCores = 1)

# run diem with fixed null polarities
fit2 <- diem(
  files = inputFile, ChosenInds = inds, ploidy = ploidies, nCores = 1,
  markerPolarity = list(c(TRUE, FALSE, TRUE))
)

## End(Not run)
```

`emPolarise` *Polarises a marker*

Description

Changes encodings of genomic markers according to user specification.

Usage

```
emPolarise(origM, changePolarity = TRUE)
```

Arguments

`origM`	A character vector of genotypes comprising of _012 encodings.
`changePolarity`	A logical scalar, indicating whether to leave the marker as is (`FALSE`) or whether to change its polarity (`TRUE`).

Value

Returns a character vector with polarised markers.

Note

Note that diem and importPolarized accept also a `U` encoding for an unknown or third allele, but `emPolarise` requires all `U` to be replaced with `_`.

See Also

`diem` for determining appropriate marker polarity with respect to a barrier to geneflow.

Examples

```
emPolarise(c("0", "0", "1", "2", "2"), TRUE)
# [1] "2" "2" "1" "0" "0"

emPolarise(c("0", "_", "2", "2", "1"), FALSE)
# [1] "0" "_" "2" "2" "1"
```

`importPolarized` *Imports genomic data polarized according to the specification*

Description

Reads genotypes from a file and changes marker polarity.

Usage

```
importPolarized(
  files,
  changePolarity,
  ChosenInds,
  ChosenSites = "all",
  nCores = 1,
  verbose = FALSE,
  ...
)
```

Arguments

`files`	A character vector with paths to files with genotypes.
`changePolarity`	A logical vector with length equal to the number of markers.
`ChosenInds`	A numeric or logical vector of indices of individuals to be included in the analysis.
`ChosenSites`	A logical vector indicating which sites are to be included in the analysis.
`nCores`	A numeric number of cores to be used for parallelisation. Must be `nCores = 1` on Windows.
`verbose`	Logical whether to show messages on import progress.
`...`	Optional numeric vector of `compartmentSizes`.

Details

For details on the input data format, check the `file` with CheckDiemFormat.

The `changePolarity` argument influences how each marker is imported. Value `FALSE` means that the marker will be imported as it is saved in the `file`. Value `TRUE` means that the genotypes encoded as `0` will be imported as `2`, and genotypes encoded in the `file` as `2` will be imported as `0`.

Value

Returns a character matrix with rows containing individual genotypes and columns containing markers.

See Also

diem for determining appropriate marker polarity with respect to a barrier to gene flow.

Examples

```
dat <- importPolarized(
  files <- system.file("extdata", "data7x3.txt", package = "diemr"),
  changePolarity = c(FALSE, TRUE, TRUE),
  ChosenInds = 1:6,
  ChosenSites = "all"
)
dat
#    m1  m2  m3
# 1 "0" "1" "2"
# 2 "0" "0" "0"
# 3 "1" "1" "0"
# 4 "1" "2" "0"
# 5 "2" "2" "1"
# 6 "2" "2" "_"
```

`ModelOfDiagnostic` *Model of Diagnostic Marker Based on All Individual State Counts*

Description

Estimates a diagnostic marker for the state counts of all genomic markers for all individuals. Using the hypothetical, diagnostic marker, calculates individual state counts with respect to their weighted similarity to the diagnostic marker states.

Usage

```
ModelOfDiagnostic(
  I4,
  OriginalHI,
  epsilon = 0.99,
  verbose = FALSE,
  folder = "likelihood",
  ...
)
```

Arguments

`I4`	a matrix or data.frame with 4 numeric columns representing character state counts for missing data, homozygots for allele 1, heterozygots, and homozygots for allele 2. Individuals in rows.
`OriginalHI`	numeric vector of length equal to number of rows in `I4`, representing hybrid indices of individuals.
`epsilon`	A numeric, specifying how much the hypothetical diagnostic markers should contribute to the likelihood calculations. Must be in `[0,1)`, keeping tolerance setting of the `R` session in mind.
`verbose`	Logical or character with path to directory where run diagnostics will be saved.

folder	character specifying path to a folder for the verbose output.
...	additional arguments.

Details

The OriginalHI can be calculated with pHetErrOnStateCount.

Value

Matrix with dimensions of I4.

See Also

diem for utilising the model to determine appropriate marker polarisation in estimating barriers to geneflow.

---

myotis *Dataset of Modified Genotypes of Bats*

Description

A subset of single nucleotide polymorphisms in *Myotis myotis* from Harazim et al. (2021). The genotypes were modified for testing purposes in such a way that markers 15 and 17 now include additional indel and substitution alleles. Eight markers used in the dataset are monomorphic.

Format

vcf file with 14 individuals and 20 markers.

Details

The data is used to test conversion of genotype data from vcf to diem format with function vcf2diem.

Source

Harazim M., Pialek L., Pikula J., Seidlova V., Zukal J., Bachorec E., Bartonicka T., Kokurewicz T., Martinkova N. (2021) Associating physiological functions with genomic variability in hibernating bats. *Evolutionary Ecology*, 35, 291-308, doi: 10.1007/s10682-020-10096-4.

Examples

```
filename <- system.file("extdata", "myotis.vcf", package = "diemr")
```

pHetErrOnStateCount *Hybrid index, heterozygosity, error rate*

Description

Using genotype allele counts, calculates the hybrid index, heterozygosity and error rate in a single individual.

Usage

```
pHetErrOnStateCount(sCount)
```

Arguments

sCount	A numeric vector of length 4 with allele counts for missing data, homozygots for allele 1, heterozygots, and homozygots for allele 2.

Details

Allele counts are genomic state counts multiplied by ploidy. As different compartments might have different ploidies (e.g. autosomal markers, sex chromosomes, mitochondrial markers), allele counts should be calculated per compartment and then summarised to obtain the correct genomic allele counts. When all individuals in each compartmenst have the same ploidy, state counts do not need to be corrected.

Value

Returns a named numeric vector with three values: p - hybrid index, Het - heterozygosity, Err - error rate.

Examples

```
pHetErrOnStateCount(sCount = c(2, 4, 2, 6))
#         p       Het       Err
# 0.5833333 0.1666667 0.1428571
```

plotDeFinetti *Plot the De Finetti Diagram for Polarized Genotypes*

Description

This function calculates genotype frequencies from polarized genotypes, ideally imported using the `importPolarized` function. It plots individuals onto a ternary De Finetti diagram and includes a curve indicating Hardy-Weinberg equilibrium if specified.

Usage

```
plotDeFinetti(
  genotypes,
  cols,
  HWE = TRUE,
  tipLabels = c("Homozygous 0", "Heterozygous 1", "Homozygous 2"),
  ...
)
```

Arguments

genotypes	A character matrix with _012 encoding of genotypes. Rows represent individuals, columns represent markers.
cols	A character vector of colors with a length equal to the number of individuals (rows) in genotypes.
HWE	Logical indicating whether to plot the curve for Hardy-Weinberg Equilibrium.
tipLabels	A character vector of length 3 with names for the ternary plot vertices.
...	additional graphical parameters (see plot.default).

Details

To import and polarize genotypes, use the importPolarized function. Alternatively, the I4 matrix can be used as input for genotypes.

Value

No return value; the function is called for its side effects.

Examples

```
gen <- importPolarized(
  file = system.file("extdata", "data7x10.txt", package = "diemr"),
  changePolarity = c(TRUE, FALSE, TRUE, TRUE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE),
  ChosenInds = 1:7
)

plotDeFinetti(gen, cols = palette.colors(nrow(gen), "Accent"), pch = 19)
```

---

plotMarkerAxis *Add a Marker Axis with Chromosome Names to a Plot of Polarized Genotypes*

Description

This function adds a marker axis with chromosome names to an existing plot of polarized genotypes. It requires that the plot is already created using plotPolarized.

Usage

```
plotMarkerAxis(
  includedSites,
  ChosenSites = "all",
  tickDist = 1e+06,
  axisInfo = NULL,
  ...
)
```

Arguments

includedSites	A character path to a file with columns CHROM and POS.
ChosenSites	A logical vector indicating which sites are to be included in the analysis.
tickDist	A numeric indicating the spacing of physical tick marks along a chromosome.
axisInfo	A list with user-defined tick positions and labels for marker axis. See Details.
...	additional arguments.

Details

Either axisInfo or includedSites must be provided. If axisInfo = NULL, the function extracts the necessary information from the includedSites file. The includedSites file should ideally be generated by vcf2diem to ensure congruence between the plotted genotypes and the respective metadata.

Tick mark distances within a chromosome are spaced at intervals of tickDist (in bp) and formatted to multiples of millions.

The positions in axisInfo (e.g., ticksPos, CHROMbreaks, CHROMnamesPos) are in units of marker index (i.e. column positions in the genotype matrix used in the previous plotPolarized call).

The optional axisInfo argument must be a list with five named elements:

- CHROMbreaks: Numeric vector of positions defining ticks that separate chromosomes.
- CHROMnamesPos: Numeric vector of label positions for chromosome names.
- CHROMnames: Character vector of chromosome names (same length as CHROMnamesPos).
- ticksPos: Numeric vector with tick positions within chromosomes.
- ticksNames: Character vector of labels corresponding to ticksPos. Potentially overlapping labels are automatically replaced with "".

The ... arguments accept additional graphical parameters passed to either axis(), mtext(), or circos.*() functions, depending on plot type. The following arguments are supported:

Rectangular axis arguments: side, col.ticks, labels, las, tick, line, tcl, cex, cex.axis, pos, outer, font, lty, lwd, lwd.ticks, hadj, padj, gap.axis, xpd.

Circular axis arguments (used in iris/circular plots): major.tick.length, lwd, labels.cex, cex, niceFacing, labels.niceFacing, facing, labels.facing, track.height, adj.

In rectangular plots, chromosome names are drawn using mtext(), with size scaled by cex * 1.1. Tick labels for positions use the same cex.

In circular plots (types "iris" or "circular"), tick positions and chromosome names are drawn using circos.axis() and circos.text(), respectively. Their sizes are controlled separately:

- `labels.cex` controls the size of tick position numbers,
- `cex` controls the size of chromosome name labels.

Similarly, text facing is controlled independently:

- `labels.niceFacing` affects the tick labels,
- `niceFacing` affects chromosome names.

The default alignment for chromosome names is set by `adj = c(1, 0.5)`, meaning right-aligned and vertically centered.

To optimize the marker axis format in iris plots, it is recommended to test with only one sample plotted to save processing time.

Value

Returns an invisible `axisInfo` list with the tick positions and labels for the marker axis.

See Also

`plotPolarized` to generate the base plot.

Examples

```
## Not run:
# Run this example in a working directory with write permissions
myo <- system.file("extdata", "myotis.vcf", package = "diemr")
vcf2diem(myo, "myo")
inds <- 1:14
fit <- diem("myo-001.txt", ChosenInds = inds, ploidy = FALSE)
gen <- importPolarized("myo-001.txt", fit$markerPolarity, inds)
h <- apply(gen, 1, function(x) pHetErrOnStateCount(sStateCount(x)))[1, ]
plotPolarized(gen, h, xlab = "")
plotMarkerAxis("myo-includedSites.txt", tickDist = 100)

plotPolarized(gen, h, type = "iris")
plotMarkerAxis("myo-includedSites.txt", tickDist = 100)

## End(Not run)
```

---

plotPolarized *Plot Polarized Genotypes*

Description

Plots genotypes that can be optionally polarized and visualized as either a rectangular or circular (iris-style) plot.

Usage

```
plotPolarized(
  genotypes,
  HI,
  cols = c("#FFFFFF", "#800080", "#FFE500", "#008080"),
  type = "rectangle",
  showProgress = FALSE,
  addMarkerAxis = FALSE,
  ...
)
```

Arguments

genotypes	A character matrix with _012 encoding of genotypes. Rows represent individuals, columns represent markers.
HI	A numeric vector of hybrid indices, one per individual (row in genotypes).
cols	A vector of four colors representing: missing data, homozygotes for 0, heterozygotes, and homozygotes for 2 (in that order).
type	Character string specifying the layout of the plot. Accepted values are "rectangle", "iris", or "circular". Partial matching is allowed.
showProgress	Logical. If TRUE, prints a percentage indicator as individuals are plotted (applies only to iris plots).
addMarkerAxis	Logical. If TRUE, a default marker axis is added below the genotype matrix (in rectangular layout) or as a radial axis (in circular layout). Requires either includedSites or axisInfo. For full customization of the axis (e.g., styling, label orientation), use plotMarkerAxis manually.
...	Additional arguments passed to internal plotting functions. See Details.

Details

To import and polarize genotypes, use importPolarized.

When using diem, the hybrid indices (HI) are saved in the file "HIwithOptimalPolarities.txt". Alternatively, you can compute HI directly from polarized genotypes (see Examples).

By default, the function plots colored tick marks for individuals, with a color change at the steepest hybrid index gradient. The second and fourth colors in cols are used for these ticks. You can:

- Disable the ticks using tick = FALSE,
- Provide a vector of tick colors (must match the number of individuals), **ordered according to** order(HI),
- Provide individual labels (e.g., accession numbers) in the same order as the rows in genotypes.

If addMarkerAxis = TRUE, you must also supply either:

- includedSites — a file path containing columns CHROM and POS, or
- axisInfo — a precomputed axis information list, as described in plotMarkerAxis.

Additional graphical parameters are passed to internal calls to `image()`, `axis()`, and `plotMarkerAxis()`. The following arguments are supported:

Image arguments: `zlim`, `xlim`, `ylim`, `add`, `xaxs`, `yaxs`, `xlab`, `ylab`, `breaks`, `useRaster`, `asp`, `cex`, `cex.lab`, `cex.main`, `cex.sub`, `axes`, `col.axis`, `cex.axis`, `family`, `font`, `font.axis`, `font.lab`, `font.main`, `font.sub`, `lab`, `xpd`.

Axis arguments: `side`, `at`, `col.ticks`, `labels`, `las`, `tick`, `line`, `pos`, `outer`, `font`, `lty`, `lwd`, `lwd.ticks`, `hadj`, `padj`, `gap.axis`, `xpd`, `cex.axis`.

Value

No return value. Called for its side effects - a visual plot of polarized genotypes. In the default color scheme:

- Purple and teal represent homozygotes (`0` and `2`),
- Yellow represents heterozygotes (`1`),
- White indicates missing or undetermined genotypes (`_`). Individuals are ordered by increasing `HI` (bottom up in rectangular plots and inside out in circular plots).

See Also

plotMarkerAxis for adding chromosome information as a custom axis to the rectangle genotype plot.

Examples

```
gen <- importPolarized(
  file = system.file("extdata", "data7x10.txt", package = "diemr"),
  changePolarity = c(TRUE, FALSE, TRUE, TRUE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE),
  ChosenInds = 1:7
)

h <- apply(gen, 1, FUN = function(x) pHetErrOnStateCount(sStateCount(x)))[1, ]

plotPolarized(genotypes = gen, HI = h)

# Incorrect tick color order
plotPolarized(gen, h, col.ticks = c(rep("purple", 5), "green", "purple"), lwd = 3)

# Correct tick color order
plotPolarized(gen, h, col.ticks = c(rep("purple", 5), "green", "purple")[order(h)], lwd = 3)

# Correct individual label order
plotPolarized(gen, h, labels = c(paste("purple", 1:5), "green 1", "purple 6"))
plotPolarized(gen, h, labels = c(paste("purple", 1:5), "green 1", "purple 6"), type = "iris")
```

rank2map *Convert SNP Ranks To Windows Corresponding to Mapping Distance*

Description

This function estimates positions of ordered single nucleotide polymorphisms (SNPs) that correspond to a window spanning a user-defined distance in the SNP positions mapped to a reference. Each window is centered at the SNP mapped position. Conversion of a SNP rank position metric to a mapped position metric is useful for kernel smoothing of the `diem` output state along a genomic sequence.

Usage

```
rank2map(includedSites, ChosenSites = "all", windowSize = 1e+07, nCores = 1)
```

Arguments

`includedSites`	A character path to a file with columns `CHROM` and `POS`.
`ChosenSites`	A logical vector indicating which sites are to be included in the analysis.
`windowSize`	A numeric window size for metric conversion in base-pairs.
`nCores`	A numeric number of cores to be used for parallelisation. Must be `nCores = 1` on Windows.

Details

Single nucleotide polymorphisms (SNPs) tend to be spread across a genome randomly. To facilitate interpretation of the `diem` output, the marker states should be assessed on the metric of their position along chromosomes (contigs). The windows for kernel smoothing might contain a variable number of markers. This function estimates which markers should be assessed together given their proximity on a chromosome.

Values in `includedSites` are in essence SNP positions in BED format with a header. The `includedSites` file should ideally be generated by vcf2diem to ensure congruence across all analyses.

The function reads SNP positions from the specified BED-like file and divides the genome into segments based on chromosomes. Each segment is then processed to identify genomic windows encompassing each SNP, considering the specified window size. This process is parallelized to enhance performance, and each SNP is considered within its chromosomal context to ensure accurate window placement.

Minimum value of `windowSize` is equal to 3, but in genomic data evaluations, window size should be at least two orders of magnitude larger. A good approximation of a useful minimum window size is $(genome size) / ((number of SNPSs) / 2)$. Throughout the diemr package, `windowSize` refers to the genomic context of the respective SNP that the user wishes to consider when smoothing over the polarized genomic states.

Value

A two-column matrix with the number of rows corresponding to the number of `ChosenSites`, indicating start and end indices of adjacent markers that are within an interval of length `windowSize` centered on the specific marker.

Note

The unit of parallelization when using `nCores > 1` is set per chromosome. This may differ from the parallelization approach used in diem, where processing of compartment files is parallelized. Note that while compartment files can correspond to chromosomes, this is not necessarily the case.

Author(s)

Natalia Martinkova

Filip Jagos 521160@mail.muni.cz

See Also

smoothPolarizedGenotypes

Examples

```
## Not run:
# Run this example in a working directory with write permissions
myo <- system.file("extdata", "myotis.vcf", package = "diemr")
vcf2diem(myo, "myo")
rank2map("myo-includedSites.txt", windowSize = 50)

## End(Not run)
```

`smoothPolarizedGenotypes`

Smooth Polarized Genotype States

Description

This function smooths polarized genotype states using a Laplace kernel density estimation. It calculates a smoothed version of the genotype states over specified physical extent of genomic content around a site. At each genomic position, the function returns a weighted mode of the genomic state.

Usage

```
smoothPolarizedGenotypes(
  genotypes,
  includedSites,
  ChosenSites = "all",
  windows = NULL,
```

```
  windowSize = 250000,
  ...
)
```

Arguments

genotypes	A character matrix with _012 encoding of genotypes. Rows represent individuals, columns represent markers.
includedSites	A character path to a file with columns CHROM and POS.
ChosenSites	A logical vector indicating which sites are to be included in the analysis.
windows	A two-column numeric matrix with indices of start and end positions for windows for all markers indicated by ChosenSites. If windows = NULL, the function calculates the windows using rank2map.
windowSize	A numeric window size for metric conversion in base-pairs.
...	Additional arguments to be passed to rank2map if windows = NULL.

Details

Ensure that ChosenSites match those used to import polarized genotypes.

The function uses a truncated and scaled Laplace kernel to weight the genotype states within a window around each marker position, based on physical positions of the markers.

The Laplace kernel weights are calculated for physical positions of the sites centered at the site being smoothed as:

$$\frac{10}{19}\exp\left(\frac{-x}{b}\right),$$

when $x < 0$, meaning that the site x is upstream of the site being smoothed, and as:

$$\frac{10}{19}\exp\left(\frac{x}{b}\right),$$

when $x \geq 0$, meaning that the site is downstream. The value x is the position of a neighbouring site relative to the site being smoothed, and b is the scale parameter of the Laplace kernel. The scale parameter is equal to:

$$b = \frac{\text{windowSize}}{2\ln(20)}.$$

See Also

rank2map

Examples

```
## Not run:
# Run this example in a working directory with write permissions
myo <- system.file("extdata", "myotis.vcf", package = "diemr")
vcf2diem(myo, "myo")
fit <- diem("myo-001.txt", ChosenInds = 1:14)
gen <- importPolarized("myo-001.txt", changePolarity = fit$markerPolarity, ChosenInds = 1:14)
```

```
h <- apply(gen, 1, \(x) pHetErrOnStateCount(sStateCount(x)))[1, ]
gen2 <- smoothPolarizedGenotypes(genotypes = gen,
   includedSites = "myo-includedSites.txt", windowSize = 50)
plotPolarized(gen, h)
plotPolarized(gen2, h)

## End(Not run)
```

`sStateCount` *Count states in a vector*

Description

Counts genomic states in one sample.

Usage

```
sStateCount(s)
```

Arguments

`s`	A character vector with elements "_", "0", "1", "2" representing missing data, homozygots for allele 1, heterozygots, and homozygots for allele 2. The vector should represent a single individual.

Details

Summarizes the number of markers that are fixed for an allele in the genome of one individual. This is used to construct the I4 matrix in diem.

Value

Numeric vector of length 4 with counts of "_", "0", "1", "2" respectively.

See Also

emPolarise for changing marker polarity.

Examples

```
genotype <- c("0", "0", "_", "2", "1", "0", "1")
sStateCount(genotype)
# [1] 1 3 2 1

# calculate state counts for a polarised genotype
sStateCount(emPolarise(genotype, TRUE))
# [1] 1 1 2 3
```

testdata *Dataset of Fish Genotypes*

Description

A subset of single nucleotide polymorphisms in fish for testing purposes of multiallelic markers.

Format

vcf file with 92 individuals and 6 markers.

Details

The data is used to test conversion of genotype data from vcf to diem format with the function `vcf2diem`.

Examples

```
filename <- system.file("extdata", "testdata.vcf", package = "diemr")
```

variantSites *Identify Variant Sites in Genotype Files*

Description

This function processes genotype data from multiple files to identify variant markers based on homozygosity thresholds for a set of individuals. It supports parallel processing to improve performance when handling large datasets.

Usage

```
variantSites(
  files,
  filename = "variantSites.txt",
  ChosenInds = "all",
  requireHomozygous = TRUE,
  nCores = 1
)
```

Arguments

files	A character vector with paths to files with genotypes.
filename	A character vector with a path where to save the converted genotypes.
ChosenInds	A numeric or logical vector of indices of individuals to be included in the analysis.
requireHomozygous	A logical or numeric vector indicating whether to require the marker to have at least one or more homozygous individual(s) for each allele.
nCores	A numeric number of cores to be used for parallelisation. Must be nCores = 1 on Windows.

Details

The results are written to a specified output file and also returned as a logical vector.

A marker is considered a variant if at least requireHomozygous individuals are homozygous for each of the two alleles encoded in the diem-formatted input files.

Parallel processing when nCores > 1 is available only for non-Windows operation Windows computers must use nCores = 1. systems.

Value

A logical vector indicating whether each marker in the dataset is a variant site (TRUE) or not (FALSE). The same results are also written to the specified output file.

Examples

```
# Run this example in a folder with write permission
files <- c(system.file("extdata", "data7x3.txt", package = "diemr"),
           system.file("extdata", "data7x10.txt", package = "diemr"))
## Not run:

variant1 <- variantSites(files, filename = "v1.txt")
variant2 <- variantSites(files, filename = "v2.txt", requireHomozygous = 2)

## End(Not run)
```

---

vcf2diem *Convert vcf files to diem format*

Description

Reads vcf files and writes genotypes of the most frequent alleles based on chromosome positions to diem format.

Usage

```
vcf2diem(
  SNP,
  filename,
  chunk = 1L,
  requireHomozygous = TRUE,
  ChosenInds = "all"
)
```

Arguments

SNP	A character vector with a path to the '.vcf' or '.vcf.gz' file, or an vcfR object. Diploid data are currently supported.
filename	A character vector with a path where to save the converted genotypes.
chunk	Numeric indicating by how many markers should the result be split into separate files.
requireHomozygous	A logical or numeric vector indicating whether to require the marker to have at least one or more homozygous individual(s) for each allele.
ChosenInds	A numeric or logical vector of indices of individuals to be included in the analysis.

Details

Importing vcf files larger than 1GB, and those containing multiallelic genotypes is not recommended. Instead, use the path to the vcf file in SNP. vcf2diem then reads the file line by line, which is a preferred solution for data conversion, especially for very large and complex genomic datasets.

The number of files vcf2diem creates depends on the chunk argument and class of the SNP object.

- Values of chunk < 100 are interpreted as the number of files into which to split data in SNP. For SNP object of class vcfR, the number of markers per file is calculated from the dimensions of SNP. When class of SNP is character, the number of markers per file is approximated from a model with a message. If this number of markers per file is inappropriate for the expected output, provide the intended number of markers per file in chunk greater than 100 (values greater than 10000 are recommended for genomic data). vcf2diem will scan the whole input specified in the SNP file, creating additional output files until the last line in SNP is reached.
- Values of chunk >= 100 mean that each output file in diem format will contain chunk number of lines with the data in SNP.

When the vcf file contains markers not informative for genome polarisation, those are removed and listed in a file ending with *omittedSites.txt* in the directory specified in the SNP argument or in the working directory. The omitted loci are identified by their information in the CHROM and POS columns, and include the QUAL column data. The last column is an integer specifying the reason why the respective marker was omitted. The reasons why markers are not informative for genome polarisation using diem are:

1. Marker has fewer than 2 alleles representing substitutions.

2. Required homozygous individuals for the 2 most frequent alleles are not present (optional, controlled by the `requireHomozygous` argument).
3. The second most frequent allele is found only in one heterozygous individual.
4. Dataset is invariant for the most frequent allele.
5. Dataset is invariant for the allele listed as the first ALT in the vcf input.

The CHROM, POS, and QUAL information for loci included in the converted files are listed in the file ending with *includedSites.txt*. Additional columns show which allele is encoded as 0 in its homozygous state and which is encoded as 2.

Value

No value returned, called for side effects.

Author(s)

Natalia Martinkova

Filip Jagos 521160@mail.muni.cz

Jachym Postulka 506194@mail.muni.cz

Examples

```
## Not run:
# vcf2diem will write files to a working directory or a specified folder
# make sure the working directory or the folder are at a location with write permission
myofile <- system.file("extdata", "myotis.vcf", package = "diemr")

vcf2diem(SNP = myofile, filename = "test1")
vcf2diem(SNP = myofile, filename = "test2", chunk = 3)

## End(Not run)
```

Index