Package: MLMOI (via r-universe)

October 31, 2024

Type Package

Title Estimating Frequencies, Prevalence and Multiplicity of Infection

Version 0.1.2

Maintainer Meraj Hashemi <meraj.hashemi.esh@gmail.com>

Description The implemented methods reach out to scientists that seek to estimate multiplicity of infection (MOI) and lineage (allele) frequencies and prevalences at molecular markers using the maximum-likelihood method described in Schneider (2018) <doi:10.1371/journal.pone.0194148>, and Schneider and Escalante (2014) <doi:10.1371/journal.pone.0097899>. Users can import data from Excel files in various formats, and perform maximum-likelihood estimation on the imported data by the package's moimle() function.

Depends R (>= 4.3.0)

Imports openxlsx (>= 4.2.5.2), Rdpack (>= 2.6), Rmpfr (>= 0.9-3),

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

RdMacros Rdpack

Repository https://mhashemihsmw.r-universe.dev

RemoteUrl https://github.com/mhashemihsmw/mlmoi

RemoteRef HEAD

RemoteSha bf3957cfaf0fbceb4204e5c61b421da681176bc4

Contents

`MLMOI` *MLMOI: An R Package to preprocess molecular data and derive prevalences, frequencies and multiplicity of infection (MOI)*

Description

The MLMOI package provides three functions:

- moimport();
- moimle();
- moimerge().

Details

The package reaches out to scientists that seek to estimate MOI and lineage frequencies at molecular markers using the maximum-likelihood method described in (Schneider 2018), (Schneider and Escalante 2018) and (Schneider and Escalante 2014). Users can import data from Excel files in various formats, and perform maximum-likelihood estimation on the imported data by the package's `moimle()` function.

Types of molecular data

Molecular data can be of types:

- microsatellite repeats (STRs);
- single nucleotide polymorphisms (SNPs);
- amino acids;
- codons (base triplets).

Import function

The function `moimport()`, is designed to import molecular data. It imports molecular data in various formats and transforms them into a standard format.

Merging Datasets

Two datasets in standard format can be merged with the function `moimerge()`.

Estimation MOI and frequencies

The function `moimle()` is designed to derive MLE from molecular data in standard format.

References

Schneider KA (2018). "Large and finite sample properties of a maximum-likelihood estimator for multiplicity of infection." *PLOS ONE*, **13**(4), 1-21. doi:10.1371/journal.pone.0194148.

Schneider KA, Escalante AA (2018). "Correction: A Likelihood Approach to Estimate the Number of Co-Infections." *PLOS ONE*, **13**(2), 1-3. doi:10.1371/journal.pone.0192877.

Schneider KA, Escalante AA (2014). "A Likelihood Approach to Estimate the Number of Co-Infections." *PLoS ONE*, **9**(7), e97899. http://dx.doi.org/10.1371%2Fjournal.pone.0097899.

`moimerge` *Merges two molecular datasets.*

Description

The function is designed to merge two datasets from separate Excel files. The data in each Excel file is placed in the first worksheet.

Usage

```
moimerge(
  file1,
  file2,
  nummtd1,
  nummtd2,
  keepmtd = FALSE,
  export = NULL,
  keepwarnings = NULL
)
```

Arguments

`file1`	string; specifying the path of the first dataset.
`file2`	string; specifying the path of the second dataset.
`nummtd1`	numeric; number of metadata columns (see `moimport()`) in the first file (default as `0`).
`nummtd2`	numeric; number of metadata columns (see `moimport()`) in the second file (default as `0`).
`keepmtd`	logical; determining whether metadata (e.g., date) should be retained (default as `TRUE`).
`export`	string; the path where the data is stored.
`keepwarnings`	string; the path where the warnings are stored.

Details

The two datasets should be already in standard format (see `moimport()`). The datasets are placed in the first worksheet of the two different Excel files. Notice that marker labels (=column labels) need to be unique.

Value

The output is a dataset in standard format which constitutes of an assembly of the input datasets.

Warnings

Warnings are generated if potential inconsistencies are detected. E.g., if the same sample occurs in both datasets and have contradicting metadata entries. The function only prints the first 50 warnings. If the number of warnings are more than 50, the user is recommended to set the argument `keepwarnings`, in order to save the warnings in an Excel file.

See Also

To import and transform data into standard format, please see the function `moimport()`.

Examples

```
#The datasets 'testDatamerge1.xlsx' and 'testDatamerge1.xlsx' are already in standard format:

infile1 <- system.file("extdata", "testDatamerge1.xlsx", package = "MLMOI")
infile2 <- system.file("extdata", "testDatamerge2.xlsx", package = "MLMOI")
outfile <- moimerge(infile1, infile2, nummtd1 = 1, nummtd2 = 2, keepmtd = TRUE)
```

---

moimle *Estimates prevalences, frequency spectra and MOI parameter.*

Description

`moimle()` derives the maximum-likelihood estimate (MLE) of the MOI parameter (Poisson parameter) and the lineage (allele) frequencies for each molecular marker in a dataset. Additionally, the lineage prevalence counts are derived.

Usage

```
moimle(file, nummtd = 0, bounds = c(NA, NA))
```

Arguments

`file`	string or data.frame; if file is a path it must specify the path to the file to be imported. The dataset can also be a data.frame object in R. The dataset must be in standard format (see `moimport()`). The first column must contain sample IDs. Adjacent columns can contain metadata, followed by columns corresponding to molecular markers.
`nummtd`	numeric; number of metadata columns (e.g. date, sample location, etc.) in the dataset (default value is `nummtd = 0`).
`bounds`	numeric vector; a vector of size 2, specifying a lower bound (1st element) and an upper bound (2nd element) for the MOI parameter. The function derives lineage frequency ML estimates by profiling the likelihood function on one of the bounds. For a marker without sign of super-infections, the lower bound is employed. If one allele is contained in every sample, the upper bound is employed.

Details

`moimle()` requires a dataset in standard format which is free of typos (e.g. incompatible and unidentified entries). Therefore, users need to standardize the dataset by employing the `moimport()` function.

If one or more molecular markers contain pathological data, the ML estimate for the Poisson parameter is either 0 or does not exist. Both estimates are meaningless, however, in the former case frequency estimates exist while they do not in the later. By setting the option `bounds` as a range for MOI parameter λ. i.e., `bounds = c(`$<\lambda_m in>$, $<\lambda_m ax>$`)`, this problem is bypassed and the ML estimates are calculated by profiling at $\lambda_m in$ or $\lambda_m ax$. If no super-infections are observed at a marker, `moimle()` uses $\lambda_m in$ as the MOI parameter estimate, $\lambda_m ax$ if one lineage is present in all samples. For regular data, the profile-likelihood estimate using $\lambda_m in$ or $\lambda_m ax$ is returned depending on whether the ML estimate falls below $\lambda_m in$ or above $\lambda_m ax$.

Value

`moimle()` returns a nested list, where the outer elements correspond to molecular markers in the dataset. The inner elements for each molecular marker contain the following information:

1. sample size,
2. allele prevalence counts,
3. observed prevalences
4. log likelihood at MLE,
5. maximum-likelihood estimate of MOI parameter,
6. maximum-likelihood estimates of lineage frequencies.

Warnings

Warnings are issued, if data is pathological at one or multiple markers. If the option `bounds` is set, but MLE of MOI parameter at a molecular marker takes a lower or higher value than $\lambda_m in or \lambda_m ax$ respectively, a warning is generated.

See Also

To import and transform data to standard format, please see the function `moimport()`.

Examples

```
#basic data analysis
infile1 <- system.file("extdata", "testDatamerge1.xlsx", package = "MLMOI")
mle1 <- moimle(infile1, nummtd = 1)
```

`moimport` *Imports molecular data in various formats and transforms them into a standard format.*

Description

`moimport()` imports molecular data from Excel workbooks. The function handles various types of molecular data (e.g. STRs, SNPs), codings (e.g. 4-letter vs. IUPAC format for SNPs), and detects inconsistencies (e.g. typos, incorrect entries). `moimport()` allows users to import data from single or multiple worksheets.

Usage

```
moimport(
  file,
  multsheets = FALSE,
  nummtd = 0,
  molecular = "str",
  coding = "integer",
  transposed = FALSE,
  keepmtd = FALSE,
  export = NULL,
  keepwarnings = NULL
)
```

Arguments

`file`	string; specifying the path to the file to be imported.
`multsheets`	logical; indicating whether data is contained in a single or multiple worksheets. The default value is `multsheets = FALSE`, corresponding to data contained in a single worksheet.
`nummtd`	numeric number or vector; number of metadata columns (e.g. date, sample location, etc.) in the worksheet(s) to be imported (default value `nummtd = 0`). In case of multiple worksheet dataset, if all worksheets have the same number of metadata columns an integer value is sufficient. If the numbers differ, they have to be specified by an integer vector.

`molecular`	string vector or list; specifies the type of molecular data to be imported. STR, SNP, amino acid and codon markers are specified with 'STR', 'SNP', 'amino' and 'codon' values, respectively (default value `molecular = 'str'`). For importing single worksheets, `molecular` is a single string or string vector. When importing multiple worksheets, `molecular` is a string in case the data contains only one type of molecular data. Else it is a list, with the k-th element being a string value or a vector describing the data types of the k-th worksheet.
`coding`	string vector or list; specifies the coding of each data variable (marker) depending on their type. Admissible values for `coding` depend on molecular data types are: 'integer', 'nearest', 'ceil' and 'floor' for STRs; SNPs with '4let' and 'iupac' for SNPs; '3let', '1let' and 'full' amino acids and 'triplet' and 'compact' for codons.
`transposed`	logical or logical vector; if markers are entered in rows and samples in columns, set `transposed = TRUE` (default value `transposed = FALSE`). When importing multiple worksheets, `transposed` can be logical vector specifying for each worksheet whether it is in transposed format.
`keepmtd`	logical; determines whether metadata (e.g., date) should be retained during import (default value `keepmtd = TRUE`).
`export`	string; the path where the imported data is stored in standardized format. Data is not stored if no path is specified (default value `export = NULL`).
`keepwarnings`	string; the path where the warnings are stored. Warnings are not stored if no path is specified (default value `keepwarnings = NULL`).

Details

Each worksheet of the data to be imported must have one of the following formats: i) one row per sample and one column per marker. Here cells can have multiple entries, separated by a special character (separator), e.g. a punctuation character. ii) one column per marker and multiple rows per sample (standard format). iii) one row per sample and multiple columns per marker. Importantly, within one worksheet formats ii) and iii) cannot be combined (see section Warnings and Errors). Combinations of other formats are permitted but might result in warnings. Additionally, Occurrence of different separators are reported (see section Warnings and Errors).

Users should check the following before data import:

- the dataset is placed in the first worksheet of the workbook;
- in case of multiple worksheets, all worksheets contain data (additional worksheets need to be removed);
- sample IDs are placed in the first column (first row in case of transposed data; see section Exceptions);
- marker labels are placed in the first row (first column in case of transposed data; see section Exceptions);
- sample IDs and as well the marker labels are unique (the duplication of ID/labels are allowed when sample/marker contains data in consecutive rows/columns);
- entries such as sentences (e.g. comments in the worksheet) or meaningless words (e.g. 'missing' for missing data) are removed from data;

- metadata columns (rows in case of transposed data) are placed between sample IDs and molecular-marker columns.

If data is contained in multiple worksheets, above requirements need to be fulfilled for every worksheet in the Excel workbook. Not all sample IDs must occur in every worksheet. The sample ID must not be confused with the patient's ID, the former refers to a particular sample taken from a patient, the latter to a unique patient. Several sample IDs can have the same patient's ID. In case of multiple-worksheet datasets, all marker labels across all worksheets need to be unique.

The option `molecular` needs to be specified as a vector, for single-worksheet data (`multsheets = FALSE`) containing different types of molecular markers. A list is specified, if data spread across multiple worksheets with different types of molecular across the worksheets. List elements are vectors or single values, referring to the types of molecular data of the corresponding worksheets. Users do not need to set a vector if all markers are of the same molecular type (single or multiple worksheet dataset).

Setting the option `coding` as vector or list is similar to setting molecular type by `molecular`. Every molecular data type has a pre-specified coding class as default which users do not need to specify. Namely, 'integer' for STRs, '4let' for SNPs, '3let' for amino acids and 'triplet' for codons.

Value

returns a data frame. `moimport()` imports heterogeneous data formats and converts them into a standard format which are free from typos (e.g. incompatible and unidentified entries) appropriate for further analyses. Metadata is retained (if `keepmtd = TRUE`) and, in case of data from multiple worksheets, unified if metadata variables have the same labels across two or more worksheets. If the argument `export` is set, then the result is saved in the first worksheet of the workbook of the file specified by `export`. The imported/exported dataset will be appropriate for other functions of the package.

Warnings and Errors

Usually warnings are generated if data is corrected pointing to suspicious entries in the original data. Users should read warnings carefully and check respective entries and apply manual corrections if necessary. In case of issues an error occurs and the function is stopped.

Usually, if arguments are not set properly, errors occur. Other cases of errors are: i) if sample IDs in a worksheet are not uniquely defined, i.e., two samples in non-consecutive rows have the same sample ID; ii) if formats 'one column per marker and multiple rows per sample' and 'one row per sample and multiple columns per marker' are mixed.

Warnings are issued in several cases. Above all, when typos (e.g., punctuation characters) are found. Entries which cannot be identified as a molecular type/coding class specified by the user are also reported (e.g., '9' is reported when marker is of type SNPs, or 'L' is reported when coding class of an amino-acid marker is '3let').

Empty rows and columns are deleted and eventually reported. Samples with ambiguous metadata (in a worksheet or across worksheets in case of multiple worksheet dataset), or missing are also reported.

The function only prints the first 50 warnings. If the number of warnings are more than 50, the user is recommended to set the argument `keepwarnings`, in order to save the warnings in an Excel file.

Exceptions

Transposed data: usually data is entered with samples in rows and markers in columns. However, on the contrary some users might enter data the opposite way. That is the case of transposed data. If so, the argument `transposed = TRUE` is set, or a vector in case of multiple worksheets with at least one worksheet being transposed.

See Also

For further details, see the following vignettes:

`vignette("dataimportcheck-list", package = "MLMOI")`

`vignette("StandardAmbiguityCodes", package = "MLMOI")`

`vignette("moimport-arguments", package = "MLMOI")`

Examples

```
#datasets are provided by the package

#importing dataset with metadata variables:
infile <- system.file("extdata", "testDatametadata.xlsx", package = "MLMOI")
moimport(infile, nummtd = 3, keepmtd = TRUE)


##more examples are included in 'examples' vignette:

#vignette("examples", package = "MLMOI")
```

Index