

Package: moimix (via r-universe)

October 9, 2026

Title Estimating multiplicity of infection from high-throughput sequencing data

Version 0.0.2.9001

Description Evaluate multiplicity of infection (MOI) of organisms using binomial mixture models and other population genetics methods. moimix can be used to simulate single nucleotide variant (SNV) data from samples with known MOI.

Depends R (>= 3.4.0), flexmix

License GPL-3 + file LICENSE

LazyData true

Imports MCMCpack, foreach, SeqArray (>= 1.12.3), SeqVarTools, scales, RColorBrewer, IRanges, S4Vectors, GenomicRanges, BiocParallel, methods, stats, graphics, utils

Suggests testthat, knitr, rmarkdown, BiocStyle

VignetteBuilder knitr

RoxygenNote 7.1.0

URL <https://github.com/bahlolab/moimix>

BugReports <https://github.com/bahlolab/moimix/issues>

Encoding UTF-8

Config/pak/sysreqs cmake make libicu-dev libx11-dev zlib1g-dev

Repository <https://plasmogenepi.r-universe.dev>

Date/Publication 2020-04-13 09:59:33 UTC

RemoteUrl <https://github.com/bahlolab/moimix>

RemoteRef HEAD

RemoteSha 802eaf1fab653690b1b1f1475c879b5189ee40ae

Contents

alleleCounts	3
bafMatrix	3
barcodes	4
binommix	4
binommixCDF	5
callMajor	5
ciMM	6
coverageBySample	6
coverageBySNP	7
dtbeta	7
dtexp	8
extractBarcode	8
extractPED	9
gambian_isolates	10
generateWindows	10
getBarcodeVariants	11
getCoordinates	11
getDosage	11
getFws	12
getGATKInfo	12
getHeterozygosity	13
getHeterozygosityBySample	13
getMAF	14
getMSE	15
getTheta	15
infomat	16
maf_summary	16
moimix	16
mse	17
perSiteAlleleCoverage	17
perSiteCoverage	18
perSiteCoverageBySample	18
perSiteCoverageBySNP	19
plot.bafMatrix	19
readEstMOI	20
rtexp	21
sampleMM	21
scaledProportion	22
seMM	22
simulateMOI	23

Index

24

alleleCounts	<i>Return allele counts from a genofile based on read counts</i>
--------------	--

Description

Return allele counts from a genofile based on read counts

Usage

```
alleleCounts(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSClass](#) object

Details

If `ref.allele = NULL`, the function returns a list of matrices corresponding to the read counts of the reference and alternate alleles respectively, while if `ref.allele = 0L` or `ref.allele = 1L` the function returns a matrix corresponding the reference or alternate read counts. Currently, the AD tag formats from varscan2re and GATK are supported.

Value

an alleleCounts matrix object, which is a list containing the

bafMatrix	<i>Compute B-allele frequency spectrum genome-wide</i>
-----------	--

Description

Compute B-allele frequency spectrum genome-wide

Usage

```
bafMatrix(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSClass](#) object

Details

Constructs a bafMatrix object which is a list consisting of a matrix of estimated B-allele frequencies and

Value

a bafMatrix object

barcodes	<i>Generate GenomicRanges object from GDS file</i>
----------	--

Description

Generate GenomicRanges object from GDS file

Usage

```
barcodes(publication)
```

Arguments

publication a character string indicating malaria publication (default Volkmann2008)

Details

Return a GenomicRanges object containing Pf barcode information from Volkmann, 2008.

binommix	<i>Fit binomial mixture model on coverage data</i>
----------	--

Description

Fit binomial mixture model on coverage data

Usage

```
binommix(counts_matrix, sample.id, k, coverage_threshold = 0L, niter = 1000)
```

Arguments

counts_matrix an [alleleCounts](#) object with ref and alt slots filled
sample.id character sample.id to fit model on.
k vector of mixture components to fit
coverage_threshold
 exclude sites with total coverage < coverage_threshold
niter number of iterations to run

binommixCDF	<i>CDF for binomial mixture model</i>
-------------	---------------------------------------

Description

CDF for binomial mixture model

Usage

```
binommixCDF(model, x, N)
```

Arguments

model	flexmix object
x	vector of read counts supporting each SNV
N	vector of coverage at SNV

Value

Vector of theoretical quantiles

callMajor	<i>Get major allele calls for isolates from BAF matrix</i>
-----------	--

Description

Extract major alleles from a GDS file using the non-reference allele frequency (NRAF) estimated from read counts directly. In the case of RAF = 0.5 we select the major allele randomly from Bernoulli distribution if use.hets is FALSE. Other wise hets are coded as 3. If get.nucleotides is TRUE a character matrix is returned with the alleles coded, otherwise if use.hets the allele is coded as N. Missing genotype calls are coded as X.

Usage

```
callMajor(gdsfile, get.nucleotides = FALSE, use.hets = FALSE)
```

Arguments

gdsfile	a SeqVarGDSCClass object
get.nucleotides	FALSE logical indicating whether to replace allele with nucleotide bases
use.hets	FALSE logical indicating whether to code hets separtely

Value

a binary matrix or character matrix containing reference and alternate alleles

ciMM	<i>Generate Confidence Intervals for parameter estimates using estimated Fisher Information Matrix</i>
------	--

Description

Generate Confidence Intervals for parameter estimates using estimated Fisher Information Matrix

Usage

```
ciMM(y, model, alpha = 0.05)
```

Arguments

y	filtered counts matrix
model	flexmix object
alpha	confidence level (default = 0.05)

Value

a data.frame containing parameter estimates and lower and upper confidence limits

coverageBySample	<i>SNP coverage rates by sample thresholds</i>
------------------	--

Description

Compute the proportion of samples within each SNP that are covered to at least threshold.

Usage

```
coverageBySample(coverage_list, threshold)
```

Arguments

coverage_list	an alleleCounts object
threshold	coverage threshold

Value

a vector of n.snps length

coverageBySNP	<i>Sample coverage rates by SNP thresholds</i>
---------------	--

Description

Compute the proportion of SNPs within each sample that are covered to at least threshold.

Usage

```
coverageBySNP(coverage_list, threshold)
```

Arguments

coverage_list	an alleleCounts object
threshold	coverage threshold

Value

a vector of n.samples length

dtbeta	<i>Truncated beta distribution</i>
--------	------------------------------------

Description

Density, distribution function, quantile function and random generation for the truncated Beta distribution with parameters shape1 and shape2 (and optional truncation bounds)

Usage

```
dtbeta(x, shape1, shape2, lower = 0, upper = 1)
ptbeta(x, shape1, shape2, lower = 0, upper = 1)
qtbeta(p, shape1, shape2, lower = 0, upper = 1)
rtbeta(n, shape1, shape2, lower = 0, upper = 1)
```

Arguments

x	vector of quantiles
shape1, shape2	non-negative parameters of Beta distributions
lower, upper	bounds for truncation
p	vector of probabilities
n	number of observations

Functions

- ptbeta: Cumulative distribution function for truncated beta
- qtbeta: Quantile function
- rtbeta: Random number generation

dtexp

Truncated exponential distrubition

Description

Density function for random generation for truncated exponential distribution

Usage

```
dtexp(x, theta, lower, upper)
```

```
lltexp(x, theta, lower, upper)
```

```
mleTexp(x, lower, upper)
```

Arguments

x	numeric vector
theta	rate parameter for exponential
lower, upper	boundary points for truncation

Functions

- lltexp: Truncated exponential distrubition
- mleTexp: Truncated exponential distrubition

extractBarcode

Extract barcode for use in COIL program

Description

Extract barcode for use in COIL program

Usage

```
extractBarcode(gdsfile, variant.id, barcode.file)
```

Arguments

gdsfile a [SeqVarGDSCClass](#) object
 variant.id a numeric vector of variant identifiers
 barcode.file character name of output file

Details

Generate SNP barcode using supplied variant ids. This is constructed by collapsing alleles over the variant ids, forming a barcode for each sample. Then a barcode file containing sample ids + barcode is generated and can be supplied to the COIL web-app or to the command line COIL program.

Value

data.frame containing barcode for each sample

Note

Need to look at how COIL constructed it's original barcode. Esp. het calls.

References

Galinsky, Kevin, et al. "COIL: a methodology for evaluating malarial complexity of infection using likelihood from single nucleotide polymorphism data." *Malaria journal* 14.1 (2015): 4.

extractPED	<i>Extract PED files from gds</i>
------------	-----------------------------------

Description

Extract PED files from gds

Usage

```
extractPED(gdsfile, moi.estimates = NULL, use.hets = FALSE, outfile = NULL)
```

Arguments

gdsfile a [SeqVarGDSCClass](#) object
 moi.estimates a vector of MOI values obtained by [binommix](#) for each sample. DEFAULT NULL
 use.hets FALSE include heterozygote genotypes
 outfile prefix of PLINK files for output (default NULL)

Details

This function writes a plink .ped and .map file for a given gdsfile. If moi.estimates is set then use.hets is redundant. It will set the sex in each the ped file to 2 if MOI > 1 and set heterozygote genotypes to missing for MOI = 1 calls. This is for use in isoRelate. If the use.hets option is true the genotypes are used as is, other wise heterzygotes are set to missing. This function is slow if there are a large number of variants in the GDS file.

gambian_isolates	<i>Plasmodium falciparum</i> SNV data from Gambia.
------------------	--

Description

Variant calls of polymorphic variants from 65 P.falciparum isolates in The Gambia, from Amambua-Ngwa A, Tetteh KK et al., 2012 <http://www.ncbi.nlm.nih.gov/pubmed/23133397>. The data was extracted from VCF files and converted to GDS format from Malaria Genomics Consortium Pf3k version 4. See the README file at ftp://ngs.sanger.ac.uk/production/pf3k/release_4/ for details. The GDS object was obtained by filtering the MalariaGen VCF files and keeping only biallelic SNPs that were polymorphic in The Gambia and passed all filters.

Format

A SeqVarGDSClazz object.

Source

ftp://ngs.sanger.ac.uk/production/pf3k/release_4/

generateWindows	<i>Construct non-overlapping Genomic Windows by Chromosome</i>
-----------------	--

Description

Construct non-overlapping Genomic Windows by Chromosome

Usage

```
generateWindows(gdsfile, window_size)
```

Arguments

gdsfile	a SeqVarGDSClazz object
window_size	size of overlapping window in base-pairs

getBarcodeVariants	<i>Helper functions for extracting variant IDs corresponding to previously published barcodes.</i>
--------------------	--

Description

Helper functions for extracting variant IDs corresponding to previously published barcodes.

Usage

```
getBarcodeVariants(gdsfile, publication = "Volkman2008")
```

Arguments

gdsfile	a SeqVarGDSCClass object
publication	a character string representing a publication

getCoordinates	<i>Helper function for producing data frame of genomic coordinates</i>
----------------	--

Description

Helper function for producing data frame of genomic coordinates

Usage

```
getCoordinates(gdsfile)
```

Arguments

gdsfile	a SeqVarGDSCClass object
---------	--

getDosage	<i>Helper function for producing matrix of allele dosages</i>
-----------	---

Description

Helper function for producing matrix of allele dosages

Usage

```
getDosage(gdsfile)
```

Arguments

gdsfile	a SeqVarGDSCClass object
---------	--

getFws	<i>Compute Fws within-host diversity statistic</i>
--------	--

Description

Compute *Fws* within-host diversity statistic

Usage

```
getFws(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSClass](#) object

Details

Compute the within host diversity statistic according to the method devised in Manske et.al, 2012. Briefly, within sample heterozygosity and within population heterozygosity are computed and assigned to ten equal sized MAF bins [0.0,0.05]...[0.45,0.5]. For each bin the mean within sample and population heterozygosity is computed. A regression line of these values through the origin is computed for each sample. The *Fws* is then $1 - \beta$.

Note

Currently only works on GATK derived gdsfiles. Needs to be updated to define populations.

References

Manske, Magnus, et al. "Analysis of Plasmodium falciparum diversity in natural infections by deep sequencing." *Nature* 487.7407 (2012): 375-379.

See Also

[getHeterozygosity](#), [getHeterozygosityBySample](#)

getGATKInfo	<i>Get GATK info tags</i>
-------------	---------------------------

Description

Obtain GATK info tags relevant to best practices.

Usage

```
getGATKInfo(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSCClass](#) object

Value

a data.frame with the following columns: chromosome position variant.id qd - (average quality by depth) mq - RMS mapping quality fs - Fisher strand bias mqrnk - MQ rank haplotypescore readposranksum

References

<http://gatkforums.broadinstitute.org/gatk/discussion/2806/howto-apply-hard-filters-to-a-call-set>

getHeterozygosity	<i>Compute population level heterozygosity</i>
-------------------	--

Description

Compute population level heterozygosity

Usage

```
getHeterozygosity(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSCClass](#) object

Note

Need to update so population ids can be submitted.

getHeterozygosityBySample	<i>Compute heterozygosity by sample</i>
---------------------------	---

Description

Compute heterozygosity by sample

Usage

```
getHeterozygosityBySample(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSCClass](#) object

Details

Follows the method outlined in Manske et al. 2012. Briefly, for each sample, the reference and alternate allele frequencies are computed as the proportion of reads covering each allele. Then heterozygosity at a SNP is $1 - (raf^2 + aaf^2)$

Value

a numeric matrix of size l by n where l is the number of samples and n is the number of SNPs.

getMAF	<i>Compute minor allele frequency</i>
--------	---------------------------------------

Description

Compute minor allele frequency

Usage

```
getMAF(gdsfile)
```

Arguments

gdsfile a [SeqVarGDSCClass](#) object

Details

Coverage based estimation of minor allele frequencies. The MAF is computed by computing the minimum of the proportion of reads covering the reference and alternate allele over all samples.

Note

Currently only supports gds files obtained from GATK callers

getMSE	<i>Model assessment for fitted model using MSE</i>
--------	--

Description

Model assessment for fitted model using MSE

Usage

```
getMSE(fitted_models)
```

Arguments

fitted_models a moimix object

getTheta	<i>Return estimated model parameters</i>
----------	--

Description

Return estimated model parameters

Usage

```
getTheta(model, k = NULL, criterion = NULL)
```

Arguments

model	stepFlexmix or flexmix object
k	if stepFlexmix choose model with k components
criterion	if stepFlexmix choose model according to information criterion

infomat	<i>Fisher Information Approximation</i>
---------	---

Description

Estimate the Fisher information matrix for a bionimal mixture model

Usage

```
infomat(y, model)
```

Arguments

y	a two column matrix of read counts
model	flexmix object

Value

A 2k by 2k Fisher information matrix

maf_summary	<i>Minor Allele Frequencies of ~ 30,000 Plasmodium falciparum SNPs.</i>
-------------	---

Description

A dataset containing information about exonic SNPs in Plasmodium falciparum from Manske et al., 2012 <http://www.ncbi.nlm.nih.gov/pubmed/22722859>.

Format

A data frame with 29621 rows and 10 variables:

Source

<http://www.malariagen.net/data>

moimix	<i>Evaluate multiplicity of infection in malaria parasites</i>
--------	--

Description

moimix enables the estimation of MOI from SNV data and to evaluate the effect of MOI on SNV data.

mse	<i>Mean Square Error for binomial mixture model</i>
-----	---

Description

Compute sum of squares for mixture components based on known truth.

Usage

```
mse(model, theta)
```

Arguments

model	estimated mixture model
theta	true mixture-model parameters of form (pi_1,...,pi_k, mu_1,...mu_k)

perSiteAlleleCoverage	<i>Compute per site per sample coverage</i>
-----------------------	---

Description

Compute per site per sample coverage

Usage

```
perSiteAlleleCoverage(gdsfile)
```

Arguments

gdsfile	a SeqVarGDSCClass object
---------	--

Value

a matrix of allele counts

perSiteCoverage	<i>Compute per variant depth accross all samples.</i>
-----------------	---

Description

Compute per variant depth accross all samples.

Usage

```
perSiteCoverage(gdsfile)
```

Arguments

gdsfile	a SeqVarGDSClass object
---------	---

Value

an integer vector

perSiteCoverageBySample	<i>Compute the proportion of samples covered up any number of bases</i>
-------------------------	---

Description

Compute the proportion of samples covered up any number of bases

Usage

```
perSiteCoverageBySample(gdsfile, threshold)
```

Arguments

gdsfile	a SeqVarGDSClass object
threshold	integer lower bound for coverage

Value

a vector of length equal to number of variants with the proportion of samples covered up to threshold

perSiteCoverageBySNP	<i>Compute proportion of SNPs covered up to a lower bound of bases within each sample.</i>
----------------------	--

Description

Compute proportion of SNPs covered up to a lower bound of bases within each sample.

Usage

```
perSiteCoverageBySNP(gdsfile, threshold)
```

Arguments

gdsfile	a SeqVarGDSCClass object
threshold	integer lower bound for coverage

Value

a vector of length equal to number of samples with the proportion of SNPs covered up to threshold

plot.bafMatrix	<i>Plot bafMatrix object</i>
----------------	------------------------------

Description

Plot bafMatrix object

Usage

```
## S3 method for class 'bafMatrix'
plot(
  x,
  sample.id,
  assignments = NULL,
  y = NULL,
  xlab = "",
  ylab = "SNV frequency",
  ylim = c(0, 1),
  pch = 16,
  ...
)
```

Arguments

<code>x</code>	a <code>bafMatrix</code> object
<code>sample.id</code>	character name of sample to plot
<code>assignments</code>	integer vector of cluster memberships (NULL)
<code>y</code>	unused argument for <code>plot.bafMatrix</code>
<code>xlab</code>	Axis label along chromosome regions
<code>ylab</code>	Axis label for SNV frequency
<code>ylim</code>	Limits for y axis default is between 0 and 1
<code>pch</code>	Plot symbol default is 16
<code>...</code>	other parameters to pass to plot

Details

Plots the genome-wide signal of MOI within an isolate from B-allele frequencies.

<code>readEstMOI</code>	<i>Parse estMOI log files in a directory into tidy data frame</i>
-------------------------	---

Description

Parse estMOI log files in a directory into tidy data frame

Usage

```
readEstMOI(path)
```

Arguments

<code>path</code>	path containing .txt files containing estMOI estimates
-------------------	--

Value

a `data.frame`

Note

Assumes that `outPrefix` set as option in `estMOI` is set to sample identifier.

rtexp	<i>Truncated exponential distribution random number generation</i>
-------	--

Description

Simulate random numbers from a truncated exponential distribution

Usage

```
rtexp(n, theta, t)
```

Arguments

n	number of random values to return
theta	rate parameter for exponential
t	cut-off threshold for truncation

Details

Simulate SNP alternate allele frequencies from a truncated exponential distribution Code is modified from that given at <https://stat.ethz.ch/pipermail/r-help/2010-June/242615.html>

Value

n random values from TEXP

sampleMM	<i>Generate iid binomial mixture random variables for testing EM implementation</i>
----------	---

Description

Generate iid binomial mixture random variables for testing EM implementation

Usage

```
sampleMM(n, N, k, mu, pi)
```

Arguments

n	number of realisations
N	number of trials
k	number of components
mu	true mixture components
pi	true mixture weights

scaledProportion	<i>sum number of samples/snps meeting coverage threshold scaled by total number of SNPs/proportions</i>
------------------	---

Description

sum number of samples/snps meeting coverage threshold scaled by total number of SNPs/proportions

Usage

```
scaledProportion(x, threshold, scale)
```

Arguments

x	integer vector of read counts
threshold	integer lower bound for coverage
scale	integer to scale results.

seMM	<i>Compute standard errors for mixture model parameter estimates</i>
------	--

Description

se is computed as the inverse of the Fisher information matrix at the EM estimates

Usage

```
seMM(y, model)
```

Arguments

y	a two column matrix of read counts
model	flexmix object

simulateMOI

*Simple simulation of MOI SNV data***Description**

Generate read count data in support of SNVs partitioned by mixtures of clones within a host. This is done by setting true clonal mixture proportions and the underlying sampling probabilities for each clone. Then a random sample of underlying SNV probabilities is sampled from either a rbeta or rtxep distribution with given parameters and read counts are generated by binomial conditional on the underlying SNV probability with the clone sampling probabilities.

Usage

```
simulateMOI(
  n.samples,
  n.snps,
  moi,
  mean_coverage,
  error_coverage,
  pi.true = NULL,
  mu.true = NULL
)
```

Arguments

n.samples	number of infected hosts in population
n.snps	number of SNPs observed
moi	number of clonal infections
mean_coverage	average coverage of reads
error_coverage	deviation in coverage of reads
pi.true	optional matrix of true mixture proportions
mu.true	optional matrix of true mixture components

Value

A list containing the following elements pi.true an n.samples by moi matrix containing true clonal proportions mu.true an n.samples by moi matrix containing true genotype proportions aaf an n.snps vector of under SNV probabilities read.counts an n.samples by n.snps matrix containing read counts supporting each SNV error.counts an n.samples by n.snps matrix containing number of error reads

Index

alleleCounts, [3](#), [4](#), [6](#), [7](#)

bafMatrix, [3](#)
barcodes, [4](#)
binommix, [4](#), [9](#)
binommixCDF, [5](#)

callMajor, [5](#)
ciMM, [6](#)
coverageBySample, [6](#)
coverageBySNP, [7](#)

dtbeta, [7](#)
dtxp, [8](#)

extractBarcode, [8](#)
extractPED, [9](#)

gambian_isolates, [10](#)
generateWindows, [10](#)
getBarcodeVariants, [11](#)
getCoordinates, [11](#)
getDosage, [11](#)
getFws, [12](#)
getGATKInfo, [12](#)
getHeterozygosity, [12](#), [13](#)
getHeterozygosityBySample, [12](#), [13](#)
getMAF, [14](#)
getMSE, [15](#)
getTheta, [15](#)

infomat, [16](#)

lltxp (dtxp), [8](#)

maf_summary, [16](#)
mleTexp (dtxp), [8](#)
moimix, [16](#)
mse, [17](#)

perSiteAlleleCoverage, [17](#)

perSiteCoverage, [18](#)
perSiteCoverageBySample, [18](#)
perSiteCoverageBySNP, [19](#)
plot, [20](#)
plot.bafMatrix, [19](#)
ptbeta (dtbeta), [7](#)

qtbeta (dtbeta), [7](#)

readEstMOI, [20](#)
rtbeta (dtbeta), [7](#)
rtexp, [21](#)

sampleMM, [21](#)
scaledProportion, [22](#)
seMM, [22](#)
SeqVarGDSCClass, [3](#), [5](#), [9–14](#), [17–19](#)
simulateMOI, [23](#)