Package 'poolHelper'

July 23, 2025

Title Simulates Pooled Sequencing Genetic Data

Version 1.1.0

Description Simulates pooled sequencing data under a variety of conditions.
Also allows for the evaluation of the average absolute difference between allele frequencies computed from genotypes and those computed from pooled data.
Carvalho et al., (2022) <doi:10.1101/2023.01.20.524733>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.1

Imports MCMCpack, Metrics, scrm, stats

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

VignetteBuilder knitr

URL https://github.com/joao-mcarvalho/poolHelper

BugReports https://github.com/joao-mcarvalho/poolHelper/issues

Config/testthat/edition 3

NeedsCompilation no

Author João Carvalho [aut, cre] (ORCID:
<https://orcid.org/0000-0002-1728-0075>),
Vítor Sousa [aut]

Maintainer João Carvalho <jgcarvalho@fc.ul.pt>

Repository CRAN

Date/Publication 2023-06-29 17:50:02 UTC

Contents

`calculatePi` *Calculate population frequency at each SNP*

Description

The frequency at a given SNP is calculated according to: `pi = c/r`, where `c` = number of minor-allele reads and r = total number of observed reads.

Usage

```
calculatePi(listPool, nLoci)
```

Arguments

`listPool`	a list containing the "minor" element, representing the number of reads with the minor-allele and the "total" element that contains information about the total number of reads. The list should also contain a "major" entry with the information about reads containing the major-allele. The output of the `poolPops` function should be used as input here.
`nLoci`	an integer that represents the total number of independent loci in the dataset.

Details

This function takes as input a list that contains the number of reads with the minor allele and the number of total reads per population at a given site. The names of the respective elements of the list should be minor and total. It works with lists containing just one set of minor and total reads, corresponding to a single locus, and with lists where each entry contains a different set of minor and total number of reads, corresponding to different loci.

Value

a list with two named entries

`pi`	a list with the allele frequencies of each population. Each list entry is a matrix, corresponding to a different locus. Each row of a matrix corresponds to a different population and each column to a different site.
`pool`	a list with three different entries: major, minor and total. This list is similar to the one obtained with the `findMinor` function.

Examples

```
# simulate coverage at 5 SNPs for two populations, assuming 20x mean coverage
reads <- simulateCoverage(mean = c(20, 20), variance = c(100, 100), nSNPs = 5, nLoci = 1)

# simulate the number of reads contributed by each individual
# for each population there are two pools, each with 5 individuals
indContribution <- popsReads(list_np = rep(list(rep(5, 2)), 2), coverage = reads, pError = 5)

# set seed and create a random matrix of genotypes for the 20 individuals - 10 per population
set.seed(10)
genotypes <- matrix(rpois(100, 0.5), nrow = 20)

# simulate the number of reference reads for the two populations
readsReference <- numberReferencePop(genotypes = genotypes, indContribution = indContribution,
size = rep(list(rep(5, 2)), 2), error = 0.01)

# create Pooled DNA sequencing data for these two populations and for a single locus
pools <- poolPops(nPops = 2, nLoci = 1, indContribution = indContribution,
readsReference = readsReference)

# define the major and minor alleles for this pool-seq data
# note that we have to select the first entry of the pools list
# because this function works for matrices
pools <- findMinor(reference = pools$reference[[1]], alternative = pools$alternative[[1]],
coverage = pools$total[[1]])

# calculate population frequency at each SNP of this locus
calculatePi(listPool = pools, nLoci = 1)
```

`computeReference` *Compute the number of reference reads over a matrix*

Description

This function works over all the rows and columns of a matrix and computes the number of reads containing the reference allele at each site and for each individual.

Usage

```
computeReference(genotypes, indContribution, error)
```

Arguments

`genotypes`	is a matrix of genotypes. Each column of the matrix should be a different site and each row a different individual. Genotypes should be encoded as 0: reference homozygote, 1: heterozygote and 2: alternative homozygote.
`indContribution`	is a matrix of individual contributions. Each row of that matrix is a different individual and each column is a different site. Thus, each entry of the matrix should contain the number of reads contributed by that individual at that particular site.
`error`	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.

Value

a matrix with the number of reference allele reads contributed by each individual. Each row of the matrix represents a different individual and each column is a different site.

Examples

```
# probability of contribution for 10 individuals at 5 sites
probs <- indProbs(np = 10, nSNPs = 5, pError = 5)

# simulate the number of reads contributed, assuming 20 coverage for each site
indContribution <- indReads(np = 10, coverage = rep(20, 5), probs = probs)

# set seed and create a random matrix of genotypes
set.seed(10)
genotypes <- matrix(rpois(50, 0.5), nrow = 10)

# simulate the number of reads with the reference allele
computeReference(genotypes = genotypes, indContribution = indContribution, error = 0.01)
```

errorHet *Average absolute difference between expected heterozygosity*

Description

Calculates the average absolute difference between the expected heterozygosity computed directly from genotypes and from pooled sequencing data.

Usage

```
errorHet(
  nDip,
  nloci,
  pools,
  pError,
  sError,
  mCov,
  vCov,
  min.minor,
  minimum = NA,
  maximum = NA,
  theta = 10
)
```

Arguments

nDip	an integer representing the total number of diploid individuals to simulate. Note that scrm::scrm() actually simulates haplotypes, so the number of simulated haplotypes is double of this.
nloci	is an integer that represents how many independent loci should be simulated.
pools	a list with a vector containing the size (in number of diploid individuals) of each pool. Thus, if a population was sequenced using a single pool, the vector should contain only one entry. If a population was sequenced using two pools, each with 10 individuals, this vector should contain two entries and both will be 10.
pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions.
sError	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.
mCov	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites.
vCov	an integer that defines the variance of the depth of coverage across all sites.

`min.minor`	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.
`minimum`	an optional integer representing the minimum coverage allowed. Sites where the population has a depth of coverage below this threshold are removed from the data.
`maximum`	an optional integer representing the maximum coverage allowed. Sites where the population has a depth of coverage above this threshold are removed from the data.
`theta`	a value for the mutation rate assuming theta = 4Nu, where u is the neutral mutation rate per locus.

Details

Different combinations of parameters can be tested to check the effect of the various parameters. The average absolute difference is computed with the mae function, assuming the expected heterozygosity computed directly from the genotypes as the `actual` input argument and the expected heterozygosity from pooled data as the `predicted` input argument.

Value

a data.frame with columns detailing the number of diploid individuals, the pool error, the number of pools, the number of individuals per pool, the mean coverage, the variance of the coverage and the average absolute difference between the expected heterozygosity computed from genotypes and from pooled data.

Examples

```
# single population sequenced with a single pool of 100 individuals
errorHet(nDip = 100, nloci = 10, pools = list(100), pError = 100, sError = 0.01,
mCov = 100, vCov = 250, min.minor = 2)

# single population sequenced with two pools, each with 50 individuals
errorHet(nDip = 100, nloci = 10, pools = list(c(50, 50)), pError = 100, sError = 0.01,
mCov = 100, vCov = 250, min.minor = 2)

# single population sequenced with two pools, each with 50 individuals
# removing sites with coverage below 10x or above 180x
errorHet(nDip = 100, nloci = 10, pools = list(c(50, 50)), pError = 100, sError = 0.01,
mCov = 100, vCov = 250, min.minor = 2, minimum = 10, maximum = 180)
```

`findMinor` *Define major and minor alleles*

Description

This function checks which of the two simulated alleles (reference or alternative) corresponds to the minor allele. This function can also be used to remove sites according to a minor-allele reads threshold.

Usage

```
findMinor(reference, alternative, coverage)
```

Arguments

`reference`	is a matrix of reference allele reads. Each row of the matrix should be a different population and each column a different site. Thus, each entry of the matrix contains the number of observed reads with the reference allele for that population at a given site.
`alternative`	is a matrix of alternative allele reads. Each row of the matrix should be a different population and each column a different site. Thus, each entry of the matrix contains the number of observed reads with the alternative allele for that population at a given site.
`coverage`	is a matrix of total coverage. Each row of the matrix should be a different population and each column a different site. Thus, each entry of the matrix contains the total number of observed reads for that population at a given site.

Details

More precisely, this function counts the number of reads with the reference or alternative allele at each site and then sets the minor allele as the least frequent of the two. This is done across all populations and so the major and minor alleles are defined at a global level. Then if the `min.minor` input is not NA, sites where the number of minor allele reads, across all populations, is below the user-defined threshold are removed.

Value

a list with three names entries

`major`	a list with one entry per locus. Each entry is a matrix with the number of major allele reads for each population. Each column represents a different site and each row a different population.
`minor`	a list with one entry per locus. Each entry is a matrix with the number of minor allele reads for each population. Each column represents a different site and each row a different population.
`total`	a list with one entry per locus. Each entry is a matrix with the coverage of each population. Each column represents a different site and each row a different population.

Examples

```
# simulate coverage at 5 SNPs for two populations, assuming 20x mean coverage
reads <- simulateCoverage(mean = c(20, 20), variance = c(100, 100), nSNPs = 5, nLoci = 1)

# simulate the number of reads contributed by each individual
# for each population there are two pools, each with 5 individuals
indContribution <- popsReads(list_np = rep(list(rep(5, 2)), 2), coverage = reads, pError = 5)

# set seed and create a random matrix of genotypes for the 20 individuals - 10 per population
set.seed(10)
genotypes <- matrix(rpois(100, 0.5), nrow = 20)

# simulate the number of reference reads for the two populations
readsReference <- numberReferencePop(genotypes = genotypes, indContribution = indContribution,
size = rep(list(rep(5, 2)), 2), error = 0.01)

# create Pooled DNA sequencing data for these two populations and for a single locus
pools <- poolPops(nPops = 2, nLoci = 1, indContribution = indContribution,
readsReference = readsReference)

# define the major and minor alleles for this Pool-seq data
# we have to select the first entry of the pools list because this function works for matrices
findMinor(reference = pools$reference[[1]], alternative = pools$alternative[[1]],
coverage = pools$total[[1]])
```

---

getNumReadsR_vector *Compute the number of reference reads*

Description

This function takes as input the total depth of coverage and computes how many of those reads are reference allele reads.

Usage

```
getNumReadsR_vector(genotype_v, readCount_v, error)
```

Arguments

genotype_v	is a vector with the genotype of a given individual. Each entry of the vector should be a different site. Genotypes should be encoded as 0: reference homozygote, 1: heterozygote and 2: alternative homozygote.
readCount_v	is a vector with the number of reads contributed by the same given individual. Each entry of that vector should be a different site.
error	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.

Details

More precisely, this function computes the number of reference reads per site for one individual, given the genotype of the individual at each site, the total number of reads observed for the individual at that site and an error rate.

Value

a vector with the number of reference allele reads. Each entry of the vector corresponds to a different individual.

Examples

```
# number of reference allele reads for three individuals, each with 10x coverage
# one individual is homozygote for the reference allele (0), other is heterozygote (1)
# and the last is homozygote for the alternative allele (2)
getNumReadsR_vector(genotype_v = c(0,1,2), readCount_v = c(10, 10, 10), error = 0.01)
```

`Ifreqs` *Compute allele frequencies from genotypes*

Description

Computes alternative allele frequencies from genotypes by dividing the total number of alternative alleles by the total number of gene copies.

Usage

```
Ifreqs(nDip, genotypes)
```

Arguments

`nDip`	an integer representing the total number of diploid individuals to simulate. Note that `scrm::scrm()` actually simulates haplotypes, so the number of simulated haplotypes is double of this.
`genotypes`	a list of simulated genotypes, where each entry is a matrix corresponding to a different locus. At each matrix, each column is a different SNP and each row is a different individual.

Value

a list of allele frequencies. Each entry of the list corresponds to a different locus.

Examples

```
genotypes <- run_scrm(nDip = 10, nloci = 10)
Ifreqs(nDip = 10, genotypes)
```

indProbs *Probability of contribution of each individual*

Description

This function computes the probability of contribution for each individual of a given pool. Please note that this function works for a single pool and should not be directly applied to situations where multiple pools were used.

Usage

```
indProbs(np, nSNPs, pError)
```

Arguments

np	an integer specifying how many individuals were pooled.
nSNPs	an integer indicating how many SNPs exist in the data.
pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal individual contribution towards the total number of reads contributed by a single pool - the higher the value the more unequal are the individual contributions.

Value

a matrix with the probabilities of contribution for each individual. Each row represents a different individual and each column is a different site.

Examples

```
# probability of contribution for 10 individuals at 5 sites
indProbs(np = 10, nSNPs = 5, pError = 100)
```

indReads *Reads contributed by each individual*

Description

This function simulates the contribution, in terms of reads, of each individual of a given pool. Please note that this function works for a single pool and should not be directly applied to situations where multiple pools were used.

Usage

```
indReads(np, coverage, probs)
```

Arguments

np	an integer specifying how many individuals were pooled.
coverage	a vector containing the total depth of coverage of a given pool. Each entry of the vector represents a different site.
probs	a matrix containing the probability of contribution of each individual. This matrix can be obtained with the indProbs function.

Value

a matrix with the number of reads contributed by each individual towards the coverage of its pool. Each row of the matrix is a different individual and each column a different site.

Examples

```
# probability of contribution for 10 individuals at 5 sites
probs <- indProbs(np = 10, nSNPs = 5, pError = 100)

# simulate the number of reads contributed, assuming 10x coverage for each site
indReads(np = 10, coverage = rep(10, 5), probs = probs)
```

---

maeFreqs *Average absolute difference between allele frequencies computed from genotypes and from Pool-seq data*

Description

Calculates the average absolute difference between the allele frequencies computed directly from genotypes and from pooled sequencing data.

Usage

```
maeFreqs(
  nDip,
  nloci,
  pError,
  sError,
  mCov,
  vCov,
  min.minor,
  minimum = NA,
  maximum = NA,
  theta = 10
)
```

Arguments

nDip	is an integer or a vector representing the total number of diploid individuals to simulate. Note that scrm::scrm() actually simulates haplotypes, so the number of simulated haplotypes is double of this. If it is a vector, then each vector entry will be simulated independently. For instance, if nDip = c(100, 200), simulations will be carried out for samples of 100 and 200 individuals.
nloci	is an integer that represents how many independent loci should be simulated.
pError	an integer or a vector representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions. If it is a vector, then each vector entry will be simulated independently.
sError	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.
mCov	an integer or a vector that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites. If it is a vector, then each vector entry will be simulated independently.
vCov	an integer or a vector that defines the variance of the depth of coverage across all sites. If the mCov is a vector, then vCov should also be a vector, with each entry corresponding to the variance of the respective entry in the mCov vector. Thus, the first entry of the vCov vector will be the variance associated with the first entry of the mCov vector.
min.minor	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.
minimum	an optional integer representing the minimum coverage allowed. Sites where the population has a depth of coverage below this threshold are removed from the data.
maximum	an optional integer representing the maximum coverage allowed. Sites where the population has a depth of coverage above this threshold are removed from the data.
theta	a value for the mutation rate assuming theta = 4Nu, where u is the neutral mutation rate per locus.

Details

The average absolute difference is computed with the mae function, assuming the frequencies computed directly from the genotypes as the actual input argument and the frequencies from pooled data as the predicted input argument.

Note that this functions allows for different combinations of parameters. Thus, the effect of different combinations of parameters on the average absolute difference can be tested. For instance, it is possible to check what is the effect of different coverages by including more than one value in the mCov input argument. This function will run and compute the average absolute difference for all combinations of the nDip, pError and mCov input arguments. This function assumes that a single pool of size nDip was used to sequence the population.

Value

a data.frame with columns detailing the number of diploid individuals, the pool error, the number of pools, the number of individuals per pool, the mean coverage, the variance of the coverage and the average absolute difference between the frequencies computed from genotypes and from pooled data.

Examples

```
# a simple test with a simple combination of parameters
maeFreqs(nDip = 100, nloci = 10, pError = 100, sError = 0.01, mCov = 100, vCov = 200, min.minor = 1)

# effect of two different pool error values in conjugation with a fixed coverage and pool size
maeFreqs(nDip = 100, nloci = 10, pError = c(100, 200), sError = 0.01,
mCov = 100, vCov = 200, min.minor = 1)

# effect of two different pool error values in conjugation with a fixed pool size
# and two different coverages
maeFreqs(nDip = 100, nloci = 10, pError = c(100, 200), sError = 0.01,
mCov = c(100, 200), vCov = c(200, 500), min.minor = 1)
```

---

maeHet *Average absolute difference between the expected heterozygosity computed from genotypes and from Pool-seq data*

Description

Calculates the average absolute difference between the expected heterozygosity computed directly from genotypes and from pooled sequencing data.

Usage

```
maeHet(
  nDip,
  nloci,
  pError,
  sError,
  mCov,
  vCov,
  min.minor,
  minimum = NA,
  maximum = NA,
  theta = 10
)
```

Arguments

nDip	is an integer or a vector representing the total number of diploid individuals to simulate. Note that scrm::scrm() actually simulates haplotypes, so the number of simulated haplotypes is double of this. If it is a vector, then each vector entry will be simulated independently. For instance, if nDip = c(100, 200), simulations will be carried out for samples of 100 and 200 individuals.
nloci	is an integer that represents how many independent loci should be simulated.
pError	an integer or a vector representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions. If it is a vector, then each vector entry will be simulated independently.
sError	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.
mCov	an integer or a vector that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites. If it is a vector, then each vector entry will be simulated independently.
vCov	an integer or a vector that defines the variance of the depth of coverage across all sites. If the mCov is a vector, then vCov should also be a vector, with each entry corresponding to the variance of the respective entry in the mCov vector. Thus, the first entry of the vCov vector will be the variance associated with the first entry of the mCov vector.
min.minor	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.
minimum	an optional integer representing the minimum coverage allowed. Sites where the population has a depth of coverage below this threshold are removed from the data.
maximum	an optional integer representing the maximum coverage allowed. Sites where the population has a depth of coverage above this threshold are removed from the data.
theta	a value for the mutation rate assuming theta = 4Nu, where u is the neutral mutation rate per locus.

Details

The average absolute difference is computed with the mae function, assuming the expected heterozygosity computed directly from the genotypes as the actual input argument and the expected heterozygosity from pooled data as the predicted input argument.

Note that this functions allows for different combinations of parameters. Thus, the effect of different combinations of parameters on the average absolute difference can be tested. For instance, it is possible to check what is the effect of different coverages by including more than one value in the mCov input argument. This function will run and compute the average absolute difference for all combinations of the nDip, pError and mCov input arguments. This function assumes that a single pool of size nDip was used to sequence the population.

Value

a data.frame with columns detailing the number of diploid individuals, the pool error, the number of pools, the number of individuals per pool, the mean coverage, the variance of the coverage and the average absolute difference between the expected heterozygosity computed from genotypes and from pooled data.

Examples

```
# a simple test with a simple combination of parameters
maeHet(nDip = 100, nloci = 10, pError = 100, sError = 0.01, mCov = 100, vCov = 200, min.minor = 1)

# effect of two different pool error values in conjugation with a fixed coverage and pool size
maeHet(nDip = 100, nloci = 10, pError = c(100, 200), sError = 0.01,
mCov = 100, vCov = 200, min.minor = 1)

# effect of two different pool error values in conjugation with a fixed pool size
# and two different coverages
maeHet(nDip = 100, nloci = 10, pError = c(100, 200), sError = 0.01,
mCov = c(100, 200), vCov = c(200, 500), min.minor = 1)
```

---

maePool *Average absolute difference between allele frequencies*

Description

Calculates the average absolute difference between the allele frequencies computed directly from genotypes and from pooled sequencing data.

Usage

```
maePool(
  nDip,
  nloci,
  pools,
  pError,
  sError,
  mCov,
  vCov,
  min.minor,
  minimum = NA,
  maximum = NA,
  theta = 10
)
```

Arguments

nDip	an integer representing the total number of diploid individuals to simulate. Note that scrm::scrm() actually simulates haplotypes, so the number of simulated haplotypes is double of this.
nloci	is an integer that represents how many independent loci should be simulated.
pools	a list with a vector containing the size (in number of diploid individuals) of each pool. Thus, if a population was sequenced using a single pool, the vector should contain only one entry. If a population was sequenced using two pools, each with 10 individuals, this vector should contain two entries and both will be 10.
pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions.
sError	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.
mCov	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites.
vCov	an integer that defines the variance of the depth of coverage across all sites.
min.minor	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.
minimum	an optional integer representing the minimum coverage allowed. Sites where the population has a depth of coverage below this threshold are removed from the data.
maximum	an optional integer representing the maximum coverage allowed. Sites where the population has a depth of coverage above this threshold are removed from the data.
theta	a value for the mutation rate assuming theta = 4Nu, where u is the neutral mutation rate per locus.

Details

Different combinations of parameters can be tested to check the effect of the various parameters. The average absolute difference is computed with the mae function, assuming the frequencies computed directly from the genotypes as the actual input argument and the frequencies from pooled data as the predicted input argument.

Value

a data.frame with columns detailing the number of diploid individuals, the pool error, the number of pools, the number of individuals per pool, the mean coverage, the variance of the coverage and the average absolute difference between the frequencies computed from genotypes and from pooled data.

Examples

```
# single population sequenced with a single pool of 100 individuals
maePool(nDip = 100, nloci = 10, pools = list(100), pError = 100, sError = 0.01,
mCov = 100, vCov = 250, min.minor = 2)

# single population sequenced with two pools, each with 50 individuals
maePool(nDip = 100, nloci = 10, pools = list(c(50, 50)), pError = 100, sError = 0.01,
mCov = 100, vCov = 250, min.minor = 2)

# single population sequenced with two pools, each with 50 individuals
# removing sites with coverage below 10x or above 180x
maePool(nDip = 100, nloci = 10, pools = list(c(50, 50)), pError = 100, sError = 0.01,
mCov = 100, vCov = 250, min.minor = 2, minimum = 10, maximum = 180)
```

---

mymae *Average absolute difference between allele frequencies computed from genotypes supplied by the user and from Pool-seq data*

Description

Calculates the average absolute difference between the allele frequencies computed directly from genotypes and from pooled sequencing data. The genotypes used should be supplied by the user and can be simulated using different software and under the demographic model of choice.

Usage

```
mymae(
  genotypes,
  pools,
  pError,
  sError,
  mCov,
  vCov,
  min.minor,
  minimum = NA,
  maximum = NA
)
```

Arguments

genotypes	a list of genotypes, where each entry is a matrix corresponding to a different locus. At each matrix, each column is a different SNP and each row is a different individual. Genotypes should be coded as 0, 1 or 2.
pools	a list with a vector containing the size (in number of diploid individuals) of each pool. Thus, if a population was sequenced using a single pool, the vector should contain only one entry. If a population was sequenced using two pools, each with 10 individuals, this vector should contain two entries and both will be 10.

pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions.
sError	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.
mCov	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites.
vCov	an integer that defines the variance of the depth of coverage across all sites.
min.minor	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.
minimum	an optional integer representing the minimum coverage allowed. Sites where the population has a depth of coverage below this threshold are removed from the data.
maximum	an optional integer representing the maximum coverage allowed. Sites where the population has a depth of coverage above this threshold are removed from the data.

Details

The average absolute difference is computed with the mae function, assuming the frequencies computed directly from the genotypes as the `actual` input argument and the frequencies from pooled data as the `predicted` input argument.

Note that this functions allows for different combinations of parameters. Thus, the effect of different combinations of parameters on the average absolute difference can be tested. For instance, it is possible to check what is the effect of different coverages by including more than one value in the `mCov` input argument. This function will run and compute the average absolute difference for all combinations of the `pools`, `pError` and `mCov` input arguments.

Value

a data.frame with columns detailing the number of diploid individuals, the pool error, the number of pools, the number of individuals per pool, the mean coverage, the variance of the coverage and the average absolute difference between the frequencies computed from genotypes and from pooled data.

Examples

```
# 100 individuals sampled at a single locus
genotypes <- run_scrm(nDip = 100, nloci = 1, theta = 5)
# compute the mean absolute error assuming a coverage of 100x and two pools of 50 individuals each
mymae(genotypes = genotypes, pools = list(c(50, 50)), pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)

# 10 individuals sampled at 5 different loci
genotypes <- run_scrm(nDip = 10, nloci = 5, theta = 5)
```

```
# compute the mean absolute error assuming a coverage of 100x and one pool of 10 individuals
mymae(genotypes = genotypes, pools = list(10), pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)
```

---

numberReferencePop *Compute the number of reference reads for multiple populations*

Description

This function computes the number of reference reads over a single locus for multiple populations.

Usage

```
numberReferencePop(genotypes, indContribution, size, error)
```

Arguments

genotypes	either a list with a single entry (one locus) or a matrix (that the function will convert to a list) containing the genotypes (coded as 0, 1 or 2). Each column of that matrix should be a different site and each row a different individual.
indContribution	a list where each entry contains the information for a single population. Each entry should be a matrix, with as many rows as the number of individuals of that population. Each row contains the number of contributed reads for a given individual and across all sites.
size	a list with one entry per population. Each entry should be a vector containing the size (in number of diploid individuals) of each pool. Thus, if a population was sequenced using a single pool, the vector should contain only one entry. If a population was sequenced using two pools, each with 10 individuals, this vector should contain two entries and both will be 10.
error	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.

Details

Note that this function will not work as intended if the input consists of multiple loci.

Value

a list with one entry per population. Each entry contains the number of reference allele reads for the individuals of that population and for that locus. Different individuals are in different rows and each columns represents a different site.

Examples

```
# simulate coverage at 5 SNPs for two populations, assuming 20x mean coverage
reads <- simulateCoverage(mean = c(20, 20), variance = c(100, 100), nSNPs = 5, nLoci = 1)

# simulate the number of reads contributed by each individual
# for each population there are two pools, each with 5 individuals
indContribution <- popsReads(list_np = rep(list(rep(5, 2)), 2), coverage = reads, pError = 5)

# set seed and create a random matrix of genotypes for the 20 individuals - 10 per population
set.seed(10)
genotypes <- matrix(rpois(100, 0.5), nrow = 20)

# simulate the number of reference reads for the two populations
numberReferencePop(genotypes = genotypes, indContribution = indContribution,
size = rep(list(rep(5, 2)), 2), error = 0.01)
```

Pfreqs *Compute allele frequencies from pooled sequencing data*

Description

Computes the frequency of the alternative allele in Pool-seq data and removes any site with too few minor-allele reads from both the pool frequencies and the frequencies computed directly from genotypes.

Usage

```
Pfreqs(reference, alternative, coverage, min.minor, ifreqs)
```

Arguments

reference	a matrix with the number of reference allele reads. Each row should be a different population and each column a different site.
alternative	a matrix with the number of alternative allele reads. Each row should be a different population and each column a different site.
coverage	a matrix with the total coverage. Each row should be a different population and each column a different site.
min.minor	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.
ifreqs	a vector of allele frequencies computed directly from the genotypes where each entry corresponds to a different site.

Details

The frequency at a given SNP is calculated according to: `pi = c/r`, where c = number of alternative allele reads and r = total number of observed reads. Additionally, if a site has less minor-allele reads than `min.minor` across all populations, that site is removed from the data.

Value

a list with two entries. The `ifreqs` entry contains the allele frequencies computed directly from genotypes and `pfreqs` the allele frequencies computed from pooled sequencing data.

Examples

```
set.seed(10)
# create a vector of allele frequencies
freqs <- runif(20)
set.seed(10)
# create a matrix with the number of reads with the alternative allele
alternative <- matrix(sample(x = c(0,5,10), size = 20, replace = TRUE), nrow = 1)
# create a matrix with the depth of coverage
coverage <- matrix(sample(100:150, size = 20), nrow = 1)
# the number of reads with the reference allele is obtained by subtracting
# the number of alternative allele reads from the depth of coverage
reference <- coverage - alternative
# compute allele frequencies from pooled sequencing data
Pfreqs(reference = reference, alternative = alternative, coverage = coverage,
min.minor = 2, ifreqs = freqs)
```

---

pool2sync *Create 'synchronized' file from Pool-seq data*

Description

Creates and saves a file with the information from Pool-seq data coded in the 'synchronized' format.

Usage

```
pool2sync(reference, alternative, file, pos = NULL)
```

Arguments

`reference`	is a list where each entry corresponds to a different locus. Each list entry is a vector with the number of reads with the reference allele. Each entry of the vector corresponds to a different SNP. This list can have a single entry if the data is comprised of a single locus.
`alternative`	is a list where each entry corresponds to a different locus. Each list entry is a vector with the number of reads with the alternative allele. Each entry of the vector corresponds to a different SNP. This list can have a single entry if the data is comprised of a single locus.

`file` is a character string naming the file to write to.

`pos` is an optional input (default is NULL). If the actual position of the SNPs are known, they can be used as input here. When working with a single locus, this should be a numeric vector with each entry corresponding to the position of each SNP. If the data has multiple loci, this should be a list where each entry is a numeric vector with the position of the SNPs for a different locus.

Details

It starts by converting the number of reads with the `reference` allele and the `alternative` allele to a A-count:T-count:C-count:G-count:N-count:deletion-count string. Here, we assume that the reference allele is always A and the alternative is always T.

Then, this A-count:T-count:C-count:G-count:N-count:deletion-count string is combined with other necessary information such as the chromosome of each SNP, the position of the SNP and the reference character. This step creates a data frame where each row corresponds to a different SNP.

A `file` is then created and saved in the current working directory, with the Pool-seq data coded in the 'synchronized' file format.

Value

a file in the current working directory containing Pool-seq data in the 'synchronized' format.

Examples

```
# simulate Pool-seq data for 100 individuals sampled at a single locus
genotypes <- run_scrm(nDip = 100, nloci = 1, theta = 5)
# simulate Pool-seq data assuming a coverage of 100x and two pools of 50 individuals each
pool <- simPoolseq(genotypes = genotypes, pools = c(50, 50), pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)
# create a 'synchronized' file of the simulated data - this will create a txt file
# pool2sync(reference = pool$reference, alternative = pool$alternative, file = "mysync.txt")

# simulate Pool-seq data for 10 individuals sampled at 5 loci
genotypes <- run_scrm(nDip = 10, nloci = 5, theta = 5)
# simulate Pool-seq data assuming a coverage of 100x and a single pool of 10 individuals
pool <- simPoolseq(genotypes = genotypes, pools = 10, pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)

# create a 'synchronized' file of the simulated data - this will create a txt file
# pool2sync(reference = pool$reference, alternative = pool$alternative, file = "mysync.txt")
```

---

pool2vcf *Create VCF file from Pool-seq data*

Description

Creates and saves a file with the information from Pool-seq data coded in the VCF format.

Usage

```
pool2vcf(reference, alternative, total, file, pos = NULL)
```

Arguments

reference	is a list where each entry corresponds to a different locus. Each list entry is a vector with the number of reads with the reference allele. Each entry of the vector corresponds to a different SNP. This list can have a single entry if the data is comprised of a single locus.
alternative	is a list where each entry corresponds to a different locus. Each list entry is a vector with the number of reads with the alternative allele. Each entry of the vector corresponds to a different SNP. This list can have a single entry if the data is comprised of a single locus.
total	is a list where each entry corresponds to a different locus. Each list entry is a vector with the total number of reads observed at each SNP. Each entry of the vector corresponds to a different SNP. This list can have a single entry if the data is comprised of a single locus.
file	is a character string naming the file to write to.
pos	is an optional input (default is NULL). If the actual position of the SNPs are known, they can be used as input here. When working with a single locus, this should be a numeric vector with each entry corresponding to the position of each SNP. If the data has multiple loci, this should be a list where each entry is a numeric vector with the position of the SNPs for a different locus.

Details

It starts by converting the number of reads with the `reference` allele, the `alternative` allele and the `total` depth of coverage to a R,A:DP string. R is the number of reads of the reference allele, A is the number of reads of the alternative allele and DP is the total depth of coverage.

Then, this information coded as R,A:DP is combined with other necessary information such as the chromosome of each SNP, the position of the SNP and the quality of the genotype among others. This creates a data frame where each row corresponds to a different SNP.

A `file` is then created and saved in the current working directory, with the header lines that go above the table in a VCF file. Finally, the data frame is appended to that file.

Value

a file in the current working directory containing Pool-seq data in the VCF format.

Examples

```
# simulate Pool-seq data for 100 individuals sampled at a single locus
genotypes <- run_scrm(nDip = 100, nloci = 1, theta = 5)
# simulate Pool-seq data assuming a coverage of 100x and two pools of 50 individuals each
pool <- simPoolseq(genotypes = genotypes, pools = c(50, 50), pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)
# create a vcf file of the simulated data - this will create a txt file
# pool2vcf(reference = pool$reference, alternative = pool$alternative,
```

```
# total = pool$total, file = "myvcf.txt")

# simulate Pool-seq data for 10 individuals sampled at 5 loci
genotypes <- run_scrm(nDip = 10, nloci = 5, theta = 5)
# simulate Pool-seq data assuming a coverage of 100x and a single pool of 10 individuals
pool <- simPoolseq(genotypes = genotypes, pools = 10, pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)

# create a vcf file of the simulated data - this will create a txt file
# pool2vcf(reference = pool$reference, alternative = pool$alternative,
# total = pool$total, file = "myvcf.txt")
```

---

poolPops *Create Pooled DNA sequencing data for multiple populations*

Description

This function combines the information for each individual of each population into information at the population level.

Usage

```
poolPops(nPops, nLoci, indContribution, readsReference)
```

Arguments

nPops	An integer representing the total number of populations in the dataset.
nLoci	An integer that represents the total number of independent loci in the dataset.
indContribution	Either a list or a matrix (when dealing with a single locus).
readsReference	A list, where each entry contains the information for a single locus. Each list entry should then have one separate entry per population. Each of these entries should be a matrix, with each row corresponding to a single individual and each column a different site. Thus, each entry of the matrix contains the number of observed reads with the reference allele for that individual at a given site. The output of the numberReference or numberReferencePop functions should be the input here.

Details

In other words, the information of all individuals in a given population is combined into a single population value and this is done for the various populations. In this situation, each entry of the indContribution and readsReference lists should contain one entry per population - being, in essence, a list within a list. Please note that this function is intended to work for multiple populations and should not be used with a single population.

Value

a list with three names entries

reference	a list with one entry per locus. Each entry is a matrix with the number of reference allele reads for each population. Each column represents a different site and each row a different population.
alternative	a list with one entry per locus. Each entry is a matrix with the number of alternative allele reads for each population. Each column represents a different site and each row a different population.
total	a list with one entry per locus. Each entry is a matrix with the coverage of each population. Each column represents a different site and each row a different population.

Examples

```
# simulate coverage at 5 SNPs for two populations, assuming 20x mean coverage
reads <- simulateCoverage(mean = c(20, 20), variance = c(100, 100), nSNPs = 5, nLoci = 1)

# simulate the number of reads contributed by each individual
# for each population there are two pools, each with 5 individuals
indContribution <- popsReads(list_np = rep(list(rep(5, 2)), 2), coverage = reads, pError = 5)

# set seed and create a random matrix of genotypes for the 20 individuals - 10 per population
set.seed(10)
genotypes <- matrix(rpois(100, 0.5), nrow = 20)

# simulate the number of reference reads for the two populations
readsReference <- numberReferencePop(genotypes = genotypes, indContribution = indContribution,
size = rep(list(rep(5, 2)), 2), error = 0.01)

# create Pooled DNA sequencing data for these two populations and for a single locus
poolPops(nPops = 2, nLoci = 1, indContribution = indContribution, readsReference = readsReference)
```

---

poolProbs *Probability of contribution of each pool*

Description

This function computes the probability of contribution of each pool towards the total depth of coverage of a single population. If multiple pools where used to sequence a single population, it is possible that some pools contribute more than others.

Usage

```
poolProbs(nPools, vector_np, nSNPs, pError)
```

Arguments

nPools	an integer indicating how many pools were used to sequence the population.
vector_np	is a vector where each entry contains the number of diploid individuals of a given pool. Thus, if a population was sequenced using two pools, each with 10 individuals, this vector would contain two entries and both will be 10.
nSNPs	an integer indicating how many SNPs exist in the data.
pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal pool contribution towards the total number of reads of a population - the higher the value the more unequal are the pool contributions.

Value

a matrix with the probabilities of contribution for each pool. Each row represents a different pool and each column is a different site.

Examples

```
# probability of contribution at 8 SNPs for 5 pools, each with 10 individuals
poolProbs(nPools = 5, vector_np = rep(10, 5), nSNPs = 8, pError = 50)
```

---

poolReads *Reads contributed by each pool*

Description

This function simulates the contribution, in terms of reads, of each pool. The number of reads contributed from all pools is equal to the total coverage of the population.

Usage

```
poolReads(nPools, coverage, probs)
```

Arguments

nPools	an integer indicating how many pools were used to sequence the population.
coverage	a vector containing the total depth of coverage of the population. Each entry of the vector represents a different site.
probs	a matrix containing the probability of contribution of each pool used to sequence the population. This matrix can be obtained with the poolProbs function.

Value

a matrix with the number of reads contributed by each pool towards the total coverage of the population. Each row of the matrix is a different pool and each column a different site.

Examples

```
# simulate the probability of contribution of each pool
probs <- poolProbs(nPools = 5, vector_np = rep(10, 5), nSNPs = 8, pError = 50)

# simulate the number of reads contributed, assuming 10x coverage for each site
poolReads(nPools = 5, coverage = rep(10, 8), probs = probs)
```

---

popReads *Compute number of reads for each individual and across all sites*

Description

This function computes the contribution of each individual towards the total coverage of a given population.

Usage

```
popReads(vector_np, coverage, pError)
```

Arguments

vector_np	is a vector where each entry contains the number of diploid individuals of a given pool. Thus, if a population was sequenced using two pools, each with 10 individuals, this vector would contain two entries and both will be 10.
coverage	a vector containing the total depth of coverage of the population. Each entry of the vector represents a different site.
pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions.

Details

If multiple pools were used to sequence a population, this will compute the contribution of each pool and then use that to calculate how many reads does that pool contribute. Next, the probability of contribution of each individual is computed and utilized to calculate the number of reads that each individual contributes towards the total number of reads observed in the corresponding pool.

Value

a matrix with the number of reads contributed by each individual. Each row of the matrix corresponds to a different individual and each column to a different site.

Examples

```
# simulate number of reads contributed by each individual towards the total population coverage
# assuming a coverage of 10x at 5 sites and two pools, each with 5 individuals
popReads(vector_np = c(5, 5), coverage = rep(10, 5), pError = 100)
```

---

popsReads *Simulate total number of reads for multiple populations*

Description

Simulates the contribution of each individual towards the total coverage of its population.

Usage

```
popsReads(list_np, coverage, pError)
```

Arguments

list_np	is a list where each entry corresponds to a different population. Each entry is a vector and each vector entry contains the number of diploid individuals of a given pool. Thus, if a population was sequenced using two pools, each with 10 individuals, this vector would contain two entries and both will be 10.
coverage	a matrix containing the total depth of coverage of all populations. Each row corresponds to a different population and each column to a different site.
pError	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions.

Details

If multiple pools were used to sequence a population, this will compute the contribution of each pool and then use that to calculate how many reads does that pool contribute. Next, the probability of contribution of each individual is computed and utilized to calculate the number of reads that each individual contributes towards the total number of reads observed in the corresponding pool. These steps will be performed for each population, thus obtaining the number of reads contributed by each individual for each population.

Value

a list with one entry per population. Each entry represents the number of reads contributed by each individual towards the total coverage of its population. Different individuals correspond to different rows and different sites to different columns.

Examples

```
# simulate coverage for two populations sequenced at 10x at 5 sites
reads <- simulateCoverage(mean = c(10, 10), variance = c(20, 20), nSNPs = 5, nLoci = 1)

# simulate the individual contribution towards that coverage
# assuming that the first population was sequenced using two pools of 5 individuals
# and the second using a single pool with 10 individuals
popsReads(list_np = list(c(5, 5), 10), coverage = reads, pError = 5)
```

---

removeSites *Apply a minor allele reads threshold*

Description

Removes sites where the total number of minor-allele reads is below a certain threshold.

Usage

```
removeSites(freqs, alternative, coverage, minor, min.minor)
```

Arguments

freqs	a vector of allele frequencies where each entry corresponds to a different site.
alternative	a matrix with the number of reads with the alternative allele. Each row should be a different population and each column a different site.
coverage	a matrix with the total coverage. Each row should be a different population and each column a different site.
minor	a matrix with the number of minor-allele reads. Each row should be a different population and each column a different site.
min.minor	is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.

Details

If a site has less minor-allele reads than `min.minor` across all populations, that site is removed from the data.

Value

a list with three named entries:

freqs	is a vector with the allele frequencies minus the frequency of the removed sites.
alternative	is a matrix with the number of alternative-allele reads per site, minus any removed sites.
coverage	is a matrix with the depth of coverage minus the coverage of the removed sites.

Examples

```
# create a vector of allele frequencies
freqs <- runif(20)

set.seed(10)
# create a matrix with the number of reads with the alternative allele
alternative <- matrix(sample(x = c(0,5,10), size = 20, replace = TRUE), nrow = 1)
# create a matrix with the depth of coverage
coverage <- matrix(sample(100:150, size = 20), nrow = 1)
# the number of reads with the reference allele is obtained by subtracting
# the number of alternative allele reads from the depth of coverage
reference <- coverage - alternative

# find the minor allele at each site
minor <- findMinor(reference = reference, alternative = alternative, coverage = coverage)
# keep only the matrix with the minor allele reads
minor <- minor[["minor"]]

# remove sites where the number of minor-allele reads is below the threshold
removeSites(freqs = freqs, alternative = alternative, coverage = coverage,
minor = minor, min.minor = 2)
```

---

remove_by_reads *Apply a coverage-based filter over a list*

Description

This function removes sites that have a coverage below a `minimum` value and sites with a coverage above a `maximum` value. This is done over multiple loci, assuming that each entry of the `reads` list is a different locus. If a list of genotypes is also supplied, then those same sites are also removed from each locus of the genotypes.

Usage

```
remove_by_reads(nLoci, reads, minimum, maximum, genotypes = NA)
```

Arguments

`nLoci`	an integer that represents how many independent loci were simulated.
`reads`	a list with the total depth of coverage. Each entry of the list should be a matrix corresponding to a different locus. Each row of that matrix should be the coverage of a different population and each column a different site.
`minimum`	an integer representing the minimum coverage allowed. Sites where any population has a depth of coverage below this threshold are removed from the data.
`maximum`	an integer representing the maximum coverage allowed. Sites where any population has a depth of coverage above this threshold are removed from the data.

genotypes — an optional list input with the genotypes. Each entry of the list should be a matrix corresponding to a different locus. Each column of the matrix should be a different site and each row a different individual.

Value

a list with the total depth of coverage similar to the reads input argument but without sites where the coverage was below the minimum or above the maximum. If the genotypes were included, a second list entry will also be included in the output, containing the genotypes minus the sites that were removed.

Examples

```
set.seed(10)

# simulate coverage for 10 locus
reads <- simulateCoverage(mean = c(25, 25), variance = c(200, 200), nSNPs = 10, nLoci = 10)

# remove sites with coverage below 10x or above 100x
reads <- remove_by_reads(nLoci = 10, reads = reads, minimum = 5, maximum = 100)
# notice that some locus no longer have 10 SNPs - those sites were removed
reads
```

---

remove_by_reads_matrix

Apply a coverage-based filter to a matrix

Description

This function removes sites that have a coverage below a minimum value and sites with a coverage above a maximum value. If a matrix of genotypes is also supplied, then those same sites are also removed from that matrix.

Usage

```
remove_by_reads_matrix(reads, minimum, maximum, genotypes = NA)
```

Arguments

reads — a matrix with the total depth of coverage. Each row of the matrix should be the coverage of a different population and each column a different site.

minimum — an integer representing the minimum coverage allowed. Sites where any population has a depth of coverage below this threshold are removed from the data.

maximum — an integer representing the maximum coverage allowed. Sites where any population has a depth of coverage above this threshold are removed from the data.

genotypes — an optional matrix input with the genotypes. Each column of the matrix should be a different site and each row a different individual.

Value

a matrix with the total depth of coverage minus the sites (i.e. columns) where the coverage for any of the populations was below the minimum or above the maximum. If genotypes were supplied, then the output will be a list, with one entry per locus. Each entry will contain the filtered coverage in the first entry and the genotypes, minus the removed sites, in the second entry.

Examples

```
set.seed(10)

# simulate coverage for a single locus - select the first entry to obtain a matrix
reads <- simulateCoverage(mean = c(25, 25), variance = c(200, 200), nSNPs = 10, nLoci = 1)[[1]]

# check the coverage matrix
reads

# remove sites with coverage below 10x or above 100x
remove_by_reads_matrix(reads = reads, minimum = 10, maximum = 100)
```

---

run_scrm *Simulate a single population*

Description

Simulates the evolution of biological sequences for a single population with variable theta values.

Usage

```
run_scrm(nDip, nloci, theta = 10)
```

Arguments

nDip	an integer representing the total number of diploid individuals to simulate. Note that scrm::scrm() actually simulates haplotypes, so the number of simulated haplotypes is double of this.
nloci	is an integer that represents how many independent loci should be simulated.
theta	a value for the mutation rate assuming theta = 4Nu, where u is the neutral mutation rate per locus.

Value

a list with genotypes. Each entry of the list corresponds to a different locus. For each locus, the genotypes are in a matrix, with each row representing a different individual and each column a different site.

Examples

```
run_scrm(nDip = 100, nloci = 10)
run_scrm(nDip = 100, nloci = 10, theta = 5)
```

`simPoolseq` *Simulate Pool-seq data*

Description

Simulates pooled sequencing data given a set of parameters and individual genotypes.

Usage

```
simPoolseq(
  genotypes,
  pools,
  pError,
  sError,
  mCov,
  vCov,
  min.minor,
  minimum = NA,
  maximum = NA
)
```

Arguments

`genotypes`	a list of genotypes, where each entry is a matrix corresponding to a different locus. At each matrix, each column is a different SNP and each row is a different individual. Genotypes should be coded as 0, 1 or 2.
`pools`	a list with a vector containing the size (in number of diploid individuals) of each pool. Thus, if a population was sequenced using a single pool, the vector should contain only one entry. If a population was sequenced using two pools, each with 10 individuals, this vector should contain two entries and both will be 10.
`pError`	an integer representing the value of the error associated with DNA pooling. This value is related with the unequal contribution of both individuals and pools towards the total number of reads observed for a given population - the higher the value the more unequal are the individual and pool contributions.
`sError`	a numeric value with error rate associated with the sequencing and mapping process. This error rate is assumed to be symmetric: error(reference -> alternative) = error(alternative -> reference). This number should be between 0 and 1.
`mCov`	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites.
`vCov`	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites.

`min.minor` is an integer representing the minimum allowed number of minor-allele reads. Sites that, across all populations, have less minor-allele reads than this threshold will be removed from the data.

`minimum` an optional integer representing the minimum coverage allowed. Sites where the population has a depth of coverage below this threshold are removed from the data.

`maximum` an optional integer representing the maximum coverage allowed. Sites where the population has a depth of coverage above this threshold are removed from the data.

Details

Note that this functions allows for different combinations of parameters. Thus, Pool-seq data can be simulated for a variety of parameters. For instance, different mean depths of coverage can be used to simulate Pool-seq data. It is also possible to simulate Pool-seq data using different pool sizes (by changing the `pools` input) and different values of the Pool-seq error parameter (`pError`).

Value

a list with three named entries:

`reference` a list with one entry per locus. Each entry is a matrix with the number of reference allele reads. Each column represents a different site.

`alternative` a list with one entry per locus. Each entry is a matrix with the number of alternative allele reads. Each column represents a different site.

`total` a list with one entry per locus. Each entry is a matrix with the total depth of coverage. Each column represents a different site.

Examples

```
# simulate Pool-seq data for 100 individuals sampled at a single locus
genotypes <- run_scrm(nDip = 100, nloci = 1, theta = 5)
# simulate Pool-seq data assuming a coverage of 100x and two pools of 50 individuals each
simPoolseq(genotypes = genotypes, pools = c(50, 50), pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)

# simulate Pool-seq data for 10 individuals sampled at 5 loci
genotypes <- run_scrm(nDip = 10, nloci = 5, theta = 5)
# simulate Pool-seq data assuming a coverage of 100x and a single pool of 10 individuals
simPoolseq(genotypes = genotypes, pools = 10, pError = 100, sError = 0.001,
mCov = 100, vCov = 250, min.minor = 0)
```

simReads *Simulate coverage at a single locus*

Description

Simulates the total number of reads, for each polymorphic site of a given locus using a negative binomial distribution.

Usage

```
simReads(mean, variance, nSNPs = NA, genotypes = NA)
```

Arguments

mean	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites. If a vector is supplied instead, the function assumes that each entry of the vector is the mean for a different population.
variance	an integer that defines the variance of the depth of coverage across all sites. If a vector is supplied instead, the function assumes that each entry of the vector is the variance for a different population.
nSNPs	an integer representing the number of polymorphic sites per locus to simulate. This is an optional input but either this or the genotypes matrix must be supplied.
genotypes	a matrix of simulated genotypes, where each column is a different SNP and each row is a different individual. This is an optional input but either this or the nSNPs must be supplied.

Details

The total number of reads is simulated with a negative binomial and according to a user-defined mean depth of coverage and variance. This function is intended to work with a matrix of genotypes, simulating the depth of coverage for each site present in the genotypes. However, it can also be used to simulate coverage distributions independent of genotypes, by choosing how many sites should be simulated (with the nSNPs option).

Value

a matrix with the total coverage per population and per site. Different rows represent different populations and each column is a different site.

Examples

```
# coverage for one population at 10 sites
simReads(mean = 20, variance = 100, nSNPs = 10)

# simulate coverage at one locus with 10 SNPs for two populations:
```

```
# the first with 100x and the second with 50x
simReads(mean = c(100, 50), variance = c(250, 150), nSNPs = 10)
```

`simulateCoverage` *Simulate total number of reads per site*

Description

This function simulates the total number of reads, for each polymorphic site using a negative binomial distribution.

Usage

```
simulateCoverage(mean, variance, nSNPs = NA, nLoci = NA, genotypes = NA)
```

Arguments

`mean`	an integer that defines the mean depth of coverage to simulate. Please note that this represents the mean coverage across all sites. If a vector is supplied instead, the function assumes that each entry of the vector is the mean for a different population.
`variance`	an integer that defines the variance of the depth of coverage across all sites. If a vector is supplied instead, the function assumes that each entry of the vector is the variance for a different population.
`nSNPs`	an integer representing the number of polymorphic sites per locus to simulate. This is an optional input but either this or the `genotypes` list must be supplied.
`nLoci`	an optional integer that represents how many independent loci should be simulated.
`genotypes`	a list of simulated genotypes, where each entry is a matrix corresponding to a different locus. At each matrix, each column is a different SNP and each row is a different individual. This is an optional input but either this or the `nSNPs` must be supplied.

Details

The total number of reads is simulated with a negative binomial and according to a user-defined mean depth of coverage and variance. This function is intended to work with a list of genotypes, simulating the depth of coverage for each site present in the genotypes. However, it can also be used to simulate coverage distributions independent of genotypes, by choosing how many loci to simulate (with the `nLoci` option) and choosing how many sites per locus should be simulated (with the `nSNPs` option).

Value

a list with the total coverage per population and per site. Each list entry is a matrix corresponding to a different locus. For each matrix, different rows represent different populations and each column is a different site.

Examples

```
# simulate 10 loci, each with 10 SNPs for a single population
simulateCoverage(mean = 100, variance = 250, nSNPs = 10, nLoci = 10)

# simulate 10 loci, each with 10 SNPs for two populations:
# the first with 100x and the second with 50x
simulateCoverage(mean = c(100, 50), variance = c(250, 150), nSNPs = 10, nLoci = 10)

# simulate coverage given a set of genotypes
# run scrm and obtain genotypes
genotypes <- run_scrm(nDip = 100, nloci = 10)
# simulate coverage
simulateCoverage(mean = 50, variance = 200, genotypes = genotypes)
```

Index