Package: nQuack (via r-universe)

October 26, 2026

Title Predicting Ploidal Level from Sequence Data

Description Predicts ploidal level from sequence data using site-based heterozygosity and a mixture models approach. See Gaynor et al. (2024) <doi:10.1002/aps3.11606>.

Version 1.0.4

Encoding UTF-8

RoxygenNote 7.3.3

Suggests knitr, dplyr, kableExtra, rmarkdown

Imports Rcpp (>= 1.0.11), RcppArmadillo (>= 14.0.2-1), truncdist, data.table, foreach, future, parallel, doParallel, extraDistr, RcppProgress, httr2, magrittr

LinkingTo Rcpp, RcppArmadillo, extraDistr, RcppProgress

SystemRequirements C++17 samtools (>= 1.10)

Depends R (>= 4.1.0)

LazyData true

License GPL (>= 2)

VignetteBuilder knitr

Maintainer Michelle L. Gaynor <shellyleegaynor@gmail.com>

URL http://mlgaynor.com/nQuack/, https://github.com/mgaynor1/nQuack/

BugReports https://github.com/mgaynor1/nQuack/issues

Config/testthat/edition 3

NeedsCompilation yes

Author Michelle L. Gaynor [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-3912-6079>)

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

Date/Publication 2026-07-16 12:50:19 UTC

RemoteUrl https://github.com/cran/nQuack

RemoteRef HEAD

RemoteSha 9f493e4ed2a3ff65a09b5e16b74cd7d61615a240

Contents

alphabetacalc . . . 2
alphabetacalctau . . . 3
alphabetacalctauvec . . . 4
alphabetacalcvec . . . 4
Bclean . . . 5
bestquack . . . 6
denoise_data . . . 7
emstepB . . . 8
emstepB3 . . . 9
emstepBB . . . 10
emstepBBU . . . 11
emstepBU . . . 12
emstepN . . . 13
emstepNA . . . 14
emstepNU . . . 15
emstepNUA . . . 16
estepB3 . . . 17
muvarcalcvec . . . 17
nQuire_reformat . . . 18
prepare_data . . . 19
process_data . . . 20
process_nquire . . . 21
process_rcpp . . . 21
quackBeta . . . 22
quackBetaBinom . . . 23
quackit . . . 25
quackNboots . . . 26
quackNormal . . . 27
quackNormalNQ . . . 28
resample_xm . . . 30
setconvert . . . 30
SetupBasicExample . . . 31
sim.ind.BB . . . 32
sim.ind.BB.tau . . . 33
sim.ind.simple . . . 35

Index **36**

`alphabetacalc` *Calculate Alpha and Beta from Mean and Variance*

Description

Calculate Alpha and Beta from Mean and Variance

Usage

```
alphabetacalc(mu, var)
```

Arguments

mu	Mean.
var	Variance.

Value

Numeric vector of alpha and beta.

Examples

```
abc <- alphabetacalc(0.5, 0.01)
```

---

alphabetacalctau *Calculate Alpha and Beta from Mean, Tau, and Error rate.*

Description

Calculate Alpha and Beta from Mean, Tau, and Error rate.

Usage

```
alphabetacalctau(mu, tau, error)
```

Arguments

mu	Mean.
tau	Overdispersion parameter. Ranges from 0 to 1, where 0 indicates less overdispersion and 1 indicates high overdispersion. Here tau must be greater than 0.
error	Sequencing error rate.

Value

Numeric vector of alpha and beta.

Examples

```
abc <- alphabetacalctau(0.5, 0.01, 0.01)
```

alphabetacalctauvec *Vector-based - Calculate Alpha and Beta from Mean, Tau, and Error rate.*

Description

Vector-based - Calculate Alpha and Beta from Mean, Tau, and Error rate.

Usage

```
alphabetacalctauvec(mu, tau, error)
```

Arguments

mu	Vector of mean.
tau	Overdispersion parameter. Ranges from 0 to 1, where 0 indicates less overdispersion and 1 indicates high overdispersion. Here tau must be greater than 0.
error	Sequencing error rate. Ranges from 0 to 1.

Value

Numeric matrix of alpha and beta.

Examples

```
abc <- alphabetacalctauvec(c(0.5,0.5), 0.01, 0.01)
```

alphabetacalcvec *Vector-based - Calculate Alpha and Beta from Mean and Variance*

Description

Vector-based - Calculate Alpha and Beta from Mean and Variance

Usage

```
alphabetacalcvec(mu, var)
```

Arguments

mu	Vector of mean.
var	Vector of variance.

Value

Numeric matrix of alpha and beta.

Examples

```
abc <- alphabetacalcvec(c(0.5, 0.5), c(0.01, 0.01))
```

`Bclean` *Remove Noise with the Beta Distribution*

Description

Here we filter allele frequencies with a beta mixture model that contains 5 mixtures: three mixtures representing cytotypes included in nQuack and two mixtures representing a U-shaped distribution. We constrained the first three mixtures to have shape and scale parameters above 1, while the last two mixtures shape and scale are constrained to be less than 1. With this implementation of expectation-maximization, we utilizes the scaled probability of each data point belonging to each mixture model to remove site where the probability of belonging to a U-shaped mixture is higher than the probability of belonging to any other mixture. Due to the computational time needed to run the expectation-maximization algorithm, by default, we simple calculate this probability matrix with the E-step and do not run the complete algorithm.

Usage

```
Bclean(xm, plot = TRUE, quick = TRUE)
```

Arguments

`xm`	Matrix with total coverage and coverage for a randomly sampled allele.
`plot`	Default to TRUE. The plots do not share the same y-axis, so careful interpretation is key. Warning, if nothing is removed, the plot of removed data will be missing.
`quick`	Default to TRUE. If set as FALSE, the expectation-maximization algorithm will be run in full.

Value

Numeric matrix with total coverage and coverage for a randomly sampled allele.

Examples

```
out <- Bclean(xm[1:100,])
```

`bestquack` *Model Selection - Expectation Maximization - Optimal Distribution and Type*

Description

This function was made to run a subset of models based on a selected distribution and type. There are many limitations to this function to make this tractable, as there are 128 models that could be run with our package. Here we do not include models or comparisons we found unhelpful, this includes the nQuire implementation and log-likelihood ratio tests.

Usage

```
bestquack(
  xm,
  distribution,
  type,
  uniform,
  mixtures = c("diploid", "triploid", "tetraploid", "hexaploid", "pentaploid"),
  samplename,
  trunc = c(0, 0),
  lowvar = FALSE,
  tau = NA,
  error = NA,
  verbose = TRUE
)
```

Arguments

`xm`	Matrix with two columns with total coverage and coverage for a randomly sampled allele.
`distribution`	May be set to normal, beta, or beta-binomial. We do not include the implementation with nQuire.
`type`	May be equal to fixed, fixed_2, or fixed_3.
`uniform`	If equal to 1, a uniform mixture is included. If equal to 0, no uniform mixture is included.
`mixtures`	Defaults to `c("diploid", "triploid", "tetraploid", "hexaploid", "pentaploid")`.
`samplename`	Name of sample to be included in output.
`trunc`	List of two values representing the lower and upper bounds for allele frequency truncation ,c_L and c_U. If allele frequency truncation was done to remove error, then you do not need to truncate the expected. If no truncation has been done, this should be set to c(0,0), which is the default.
`lowvar`	Default to FALSE. When false, variance is equal to 0.01. If set to TRUE and tau and error are not provided, the variance will be set as 0.001.

tau	Sequencing overdispersion parameter. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
error	Sequencing error rate. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
verbose	Default to TRUE. If TRUE, progress messages will be printed to the console.

Value

BIC scores and log-likelihood (LL) for the included mixture models. For BIC, the smallest score is the most likely model. For LL, the largest score is the most likely model.

Examples

```
out <- bestquack(xm[1:100,],
                 distribution = "normal",
                 type = "fixed",
                 uniform = 1,
                 samplename = "sample1")
```

---

denoise_data *Denoise Data*

Description

Here we filter allele frequencies with a normal + uniform mixture model. nQuack utilizes the scaled probability of each data point belonging to each mixture model, which is inferred in the expectation maximization algorithm. We remove allele frequencies where probability of belonging to uniform mixture is higher than their probability of belonging to any other mixture. We also implement nQuire's denoise method here, which utilizes the inferred alpha parameter and a histogram of base frequencies to filter the data.

Usage

```
denoise_data(xm, plot = TRUE, filter = "both")
```

Arguments

xm	Matrix with total coverage and coverage for a randomly sampled allele.
plot	Default to TRUE. The plots do not share the same y-axis, so careful interpretation is key. Warning, if nothing is removed, the plot of removed data will be missing.
filter	Indicates which method to remove data based upon. Options: 'both', 'nquire', or 'nquack'. nQuack utilizes the scaled probability of each data point belonging to each mixture model, removing sites where the probability of belonging to uniform mixture is higher than their probability of belonging to any other mixture. nQuire utilizes the inferred alpha parameter and a histogram of base frequencies to filter the data.

Value

Numeric matrix with total coverage and coverage for a randomly sampled allele.

Examples

```
out <- denoise_data(xm[1:100,])
```

`emstepB` *Expectation maximization - Beta Distribution*

Description

This function calculates the log-likelihood using the expectation maximization algorithm with Nelder-Mead numerical optimization and a beta distribution.

Usage

```
emstepB(parmlist, xi, niter, epsilon, trunc, type = "free")
```

Arguments

`parmlist`	A list containing initial alpha, mean, and variance values.
`xi`	List of observations, in this case allele frequencies.
`niter`	Max number of iterates.
`epsilon`	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
`trunc`	List of two values representing the lower and upper bounds, c_L and c_U.
`type`	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed-2" (estimated parameter(s): alpha and variance), or "fixed-3" (estimated parameter(s): variance). If avec is length of 1, fixed and fixed-3 will not be able to return a log-likelihood.

Value

List of elements including the log likelihood, the negative log likelihood, the number of iterates, and the optimized parameter values.

Examples

```
if(exists("crazy")){
  xi <- (xm[,2]/xm[,1])
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepB(p,
```

```
                    xi,
                    niter = 100,
                    epsilon = 0.1,
                    trunc = c(0.0,0.0))
}
```

---

emstepB3 *Expectation maximization - Beta + Beta + Beta Distribution*

Description

This function is made for the `Bclean()` function and preforms expectation maximization with Nelder-Mead numerical optimization for beta distribution.

Usage

```
emstepB3(parmlist, xi, niter, epsilon, trunc)
```

Arguments

`parmlist`	A list containing initial alpha, mean, and variance.
`xi`	Matrix where the first column is total coverage and the second is the count of base A or B.
`niter`	Max number of iterates.
`epsilon`	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
`trunc`	List of two values representing the lower and upper bounds, c_L and c_U.

Value

List of elements including the negative log likelihood, the number of iterates, and the optimized parameter values.

Examples

```
 if(exists("crazy")){
    xi <- (xm[,2]/xm[,1])
    tcalc <- alphabetacalcvec(mu = c(0.287, 0.50, 0.713),
                              var = c(0.01, 0.01, 0.01))
    set <-  list(avec = c(0.25, 0.25, 0.25, 0.125, 0.125),
                 t1vec = c(tcalc[,1], 0.5, 0.33),
                 t2vec = c(tcalc[,2], 0.33, 0.5))
    checkB <- emstepB3(set,
                       xi,
                       1000,
                       0.1,
                       c(0,0))
}
```

emstepBB *Expectation maximization - Beta-Binomial Distribution*

Description

This function calculates the negative log-likelihood using the expectation maximization algorithm with Nelder-Mead numerical optimization and beta-binomial distribution.

Usage

```
emstepBB(parmlist, xm, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance.
xm	Matrix where the first column is total coverage and the second is the count of base A or B.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating "Free" or "Fixed".

Value

List of elements including the negative log likelihood, the number of iterates, and the optimized parameter values.

Examples

```
if(exists("crazy")){
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepBB(p,
                   xm,
                   niter = 100,
                   epsilon = 0.1,
                   trunc = c(0.0,0.0))
}
```

emstepBBU *Expectation maximization - Beta-Binomial and Uniform Distributions*

Description

This function calculates the log-likelihood using the expectation-maximization algorithm with Nelder-Mead numerical optimization and beta distribution with one uniform mixture.

Usage

```
emstepBBU(parmlist, xm, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance values.
xm	Matrix where the first column is total coverage and the second is the count of base A or B.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed-2" (estimated parameter(s): alpha and variance), or "fixed-3" (estimated parameter(s): variance). If avec is length of 1, fixed and fixed-3 will not be able to return a log-likelihood.

Value

List of elements including the log likelihood, the negative log likelihood, the number of iterates, and the optimized parameter values.

Examples

```
 if(exists("crazy")){
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
          mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
          svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepBBU(p,
                    xm,
                    niter = 100,
                    epsilon = 0.1,
                    trunc = c(0.0,0.0))
}
```

emstepBU *Expectation maximization - Beta and Uniform Distributions*

Description

This function calculates the log-likelihood using the expectation maximization algorithm with Nelder-Mead numerical optimization and beta distribution with one uniform mixture.

Usage

```
emstepBU(parmlist, xi, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance values.
xi	List of observations, in this case allele frequencies.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed_2" (estimated parameter(s): alpha and variance), or "fixed_3" (estimated parameter(s): variance). If avec is length of 1, fixed and fixed_3 will not be able to return a log-likelihood.

Value

List of elements including the log likelihood, the negative log likelihood, the number of iterates, and the optimized parameter values.

Examples

```
if(exists("crazy")){
  xi <- (xm[,2]/xm[,1])
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepBU(p,
                   xi,
                   niter = 100,
                   epsilon = 0.1,
                   trunc = c(0.0,0.0))
}
```

emstepN *Expectation maximization - Normal Distribution*

Description

This function calculates the log-likelihood using the expectation maximization algorithm with the Normal Distribution. This code follows nQuire and does not use an augmented likelihood.

Usage

```
emstepN(parmlist, xi, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance values.
xi	List of observations, in this case allele frequencies.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed_2" (estimated parameter(s): alpha and variance), or "fixed_3" (estimated parameter(s): variance).

Value

List of elements including the log-likelihood, the number of iterates, and the optimized parameter values.

Examples

```
 if(exists("crazy")){
  xi <- (xm[,2]/xm[,1])
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepN(p,
                  xi,
                  niter = 100,
                  epsilon = 0.1,
                  trunc = c(0.0,0.0))
}
```

emstepNA *Expectation maximization - Normal Distribution*

Description

This function calculates the log-likelihood using the expectation maximization algorithm with the Normal Distribution. This code is not identical to nQuire and uses an augmented likelihood.

Usage

```
emstepNA(parmlist, xi, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance values.
xi	List of observations, in this case allele frequencies.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed_2" (estimated parameter(s): alpha and variance), or "fixed_3" (estimated parameter(s): variance).

Value

List of elements including the log-likelihood, the number of iterates, and the optimized parameter values.

Examples

```
if(exists("crazy")){
  xi <- (xm[,2]/xm[,1])
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepNA(p,
                   xi,
                   niter = 100,
                   epsilon = 0.1,
                   trunc = c(0.0,0.0))
}
```

emstepNU *Expectation maximization - Normal and Uniform Distribution*

Description

This function calculates the log-likelihood using the expectation maximization algorithm with the Normal-Uniform Distribution. This code follows nQuire and does not use an augmented likelihood.

Usage

```
emstepNU(parmlist, xi, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance values. The list of alpha must include a proportion for the uniform mixture.
xi	List of observations, in this case allele frequencies.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed_2" (estimated parameter(s): alpha and variance), or "fixed_3" (estimated parameter(s): variance).

Value

List of elements including the log-likelihood, the number of iterates, and the optimized parameter values.

Examples

```
 if(exists("crazy")){
  xi <- (xm[,2]/xm[,1])
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepNU(p,
                   xi,
                   niter = 100,
                   epsilon = 0.1,
                   trunc = c(0.0,0.0))
}
```

emstepNUA *Expectation maximization - Normal Distribution*

Description

This function calculates the log-likelihood using the expectation maximization algorithm with the Normal-Uniform Distribution. This code is not identical to nQuire and uses an augmented likelihood.

Usage

```
emstepNUA(parmlist, xi, niter, epsilon, trunc, type = "free")
```

Arguments

parmlist	A list containing initial alpha, mean, and variance values. The list of alpha must include a proportion for the uniform mixture.
xi	List of observations, in this case allele frequencies.
niter	Max number of iterates.
epsilon	Epsilon value for convergence tolerance. When the absolute delta log-likelihood is below this value, convergence is reached.
trunc	List of two values representing the lower and upper bounds, c_L and c_U.
type	String indicating model type. Options: "free" (estimated parameter(s): alpha, mean, and variance), "fixed" (estimated parameter(s): alpha), "fixed_2" (estimated parameter(s): alpha and variance), or "fixed_3" (estimated parameter(s): variance).

Value

List of elements including the log-likelihood, the number of iterates, and the optimized parameter values.

Examples

```
if(exists("crazy")){
  xi <- (xm[,2]/xm[,1])
  p = list(avec = c(0.11, 0.22, 0.34, 0.22, 0.11),
           mvec = c(0.20, 0.33, 0.50, 0.67, 0.80),
           svec = c(0.01, 0.01, 0.01, 0.01, 0.01));
  mout <- emstepNUA(p,
                    xi,
                    niter = 100,
                    epsilon = 0.1,
                    trunc = c(0.0,0.0))
}
```

estepB3 *E-Step for Expectation Maximization - Beta + Beta + Beta Distribution*

Description

This is used in the `Bclean()` function. Here we complete the E-Step and calculate the log-likelihood. Modifications include a correction for the truncated distribution.

Usage

```
estepB3(parmlist, xi, trunc)
```

Arguments

`parmlist`	A list containing initial alpha, mean, and variance.
`xi`	List of observations, in this case allele frequencies.
`trunc`	List of two values representing the lower and upper bounds, c_L and c_U.

Value

List of zprob, parm.list, xi, denom, and trunc.

Examples

```
if(exists("crazy")){
xi <- (xm[,2]/xm[,1])
tcalc <- alphabetacalcvec(mu = c(0.287, 0.50, 0.713),
                          var = c(0.01, 0.01, 0.01))
set <-  list(avec = c(0.25, 0.25, 0.25, 0.125, 0.125),
             t1vec = c(tcalc[,1], 0.5, 0.33),
             t2vec = c(tcalc[,2], 0.33, 0.5))
checkB <- estepB3(set,
                  xi,
                  c(0,0))
}
```

muvarcalcvec *Variance calculation from Mean, Tau, and Sequencing Error*

Description

This function is used to calculate variance.

Usage

```
muvarcalcvec(mu, tau, error)
```

Arguments

mu	Vector of means.
tau	Sequence overdispersion parameter for read counts.
error	Sequencing error rate.

Value

Mean and variance for the associated tau and error.

Examples

```
var <- muvarcalcvec(mu = 0.5, tau = 0.01, error = 0.01)
```

---

nQuire_reformat *Data Preparation - Use nQuire's Data*

Description

This function reduce a three column data frame to two columns by randomly sampling allele A or B for every site. This is used in our function `process_nquire()`

Usage

```
nQuire_reformat(xm)
```

Arguments

xm	A matrix with three columns: Total Coverage, Counts for Allele A, and Counts for Allele B.

Value

Numeric Matrix with total coverage and coverage for a randomly sampled allele.

Examples

```
 if(exists("nxm")){
  out <- nQuire_reformat(nxm)
}
```

prepare_data *Prepare Data - Step 1*

Description

This function transforms a BAM file into a text file. Specifically, this function uses samtools mpileup to translate your BAM into a tab-separated file. We then filter this file to remove indels and deletions. When running this function, a temporary folder will be created (named 'temp/'), however this folder will be removed once the process is complete.

Usage

```
prepare_data(name, inpath, outpath, tempfolder = "temp")
```

Arguments

name	File name without the suffix. For example, if your file is called "frog.bam", this input should be "frog".
inpath	Location of input file.
outpath	Location for output file.
tempfolder	Location for temp folder.

Details

Warning, due to the processing time needed for samtools mpileup, this step may take some time. This function also requires samtools to be located locally. Please see our Data Preparation article for more information. Warning, this writes a temporary folder titled 'temp'. If you want to run multiple samples at once, we suggest you set the working directory to separate locations to ensure that your temp folder/files are not overwritten.

Value

Writes text file with the following columns: chromosome, position, depth, A, C, G, and T.

Examples

```
if(exists("crazy")){
## Prepare many samples
  inpath <- "filtered/"
  outpath <- "Processed/"
  filelist <- list.files(path = inpath, pattern = "*.bam" )
  filelist <- gsub(".bam", "", filelist)
  for( i in 1:length(filelist)){
    prepare_data(filelist[i], inpath, outpath)
  }
}
```

process_data *Process Data - Step 2*

Description

Based on the file generated with `prepare_data()`, which contains the total depth and sequencing coverage for each nucleotide (A, C, G, and T), this function remove all but single nucelotide polymorphisms. When supplied, this function will filter on coverage or allele frequency. To filter by total coverage, a user must supply the `min.depth` and `max.depth.quantile.prob`. If an `error` is provided, sites will be retained where allele coverage is greater than the sequencing error rate times the total coverage, but less than one minus the sequencing error rate times the total coverage. Lastly, based on `trunc`, allele frequencies will be filtered based on a provided lower and upper bound. Finally, the function samples a single allele frequency per site to avoid data duplication.

Usage

```
process_data(
  file,
  min.depth = 2,
  max.depth.quantile.prob = 0.9,
  error = 0.01,
  trunc = c(0, 0)
)
```

Arguments

`file`	Output txt file created with `prepare_data()`.
`min.depth`	Minimum sequencing depth, default as 2.
`max.depth.quantile.prob`	Maximum sequencing depth quantile cut off, default = 0.9.
`error`	Sequencing error rate. If an `error` is provided, sites will be retained where allele coverage is greater than the sequencing error rate times the total coverage, but less than one minus the sequencing error rate times the total coverage.
`trunc`	List of two values representing the lower and upper bounds, c_L and c_U which are used to filter allele frequencies.

Value

Numeric matrix with total coverage and coverage for a randomly sampled allele.

Examples

```
if(file.exists("mybamfile.csv")){
  cleaned_data <- process_data(file = "mybamfile.csv")
}
```

process_nquire *Use nQuire's Data*

Description

If you happen to like nQuire's data preparation more than ours, uses their data in our program. After processing samples with nQuire's `create` and `view` functions, the resulting text file can be read into R. To prepare the data frame for nQuack, we reduce the three column data frame to two columns by randomly sampling allele A or B for every site.

Usage

```
process_nquire(file)
```

Arguments

`file`	Output text file created with nQuire.

Value

Numeric matrix with total coverage and coverage for a randomly sampled allele.

Examples

```
if(file.exists("mynQuirefile.bin")){
  cleaned_data <- process_nquire(file = "mynQuirefile.bin")
}
```

process_rcpp *Data Preparation - Matrix Filtering*

Description

Based on supplied matrix with total depth and sequencing coverage for each nucleotide (A, C, G, and T) this function remove all but single nucelotide polymorphisms. When supplied, this function will filter on coverage or allele frequency. Finally, the function samples a single allele frequency per site to avoid data duplication.

Usage

```
process_rcpp(x, mindepth, maxprob, trunc, error)
```

Arguments

x	Matrix with five columns: Depth, A, C, G, and T.
mindepth	Minimum depth, default = 15.
maxprob	Maximum depth quantile cut off, default = 0.9.
trunc	List of two values representing the lower and upper bounds,c_L and c_U.
error	Sequencing error rate.

Value

Numeric Matrix with total coverage and coverage for a randomly sampled allele.

Examples

```
if(exists("dfm")){
  allelefreq <- process_rcpp(dfm, min.depth, max.depth.quantile.prob, trunc, error)
}
```

---

quackBeta *Model Selection - Expectation Maximization - Beta Mixture*

Description

This function uses the expectation maximization of both the beta and beta-uniform mixture models for model selection. Here we can run up to 32 mixture models.

Usage

```
quackBeta(
  xm,
  samplename,
  cores,
  parallel = FALSE,
  trunc = c(0, 0),
  lowvar = FALSE,
  tau = NA,
  error = NA,
  free = FALSE,
  verbose = TRUE
)
```

Arguments

xm	Matrix with two columns with total coverage and coverage for a randomly sampled allele.
samplename	Name of sample to be included in output.

cores	Threads available to run process in parallel.
parallel	default = FALSE, set to true if cores > 1.
trunc	List of two values representing the lower and upper bounds for allele frequency truncation , c_L and c_U. If allele frequency truncation was done to remove error, then you do not need to truncate the expected. If no truncation has been done, this should be set to c(0,0), which is the default.
lowvar	Default to FALSE. When false, variance is equal to 0.01. If set to TRUE and tau and error are not provided, the variance will be set as 0.001.
tau	Sequencing overdispersion parameter. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
error	Sequencing error rate. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
free	default = FALSE, skip the free model calculation and does not calculate delta log-likelihood.
verbose	Default to TRUE. If TRUE, progress messages will be printed to the console.

Value

BIC scores and log-likelihood (LL) mixture models including diploid, triploid, tetraploid, pentaploid, and hexaploid. When free = TRUE, the delta log-likelihood (dLL) is calculated based on the associated free model (without or with a uniform mixture). For BIC or delta-log likelihood, the smallest score is the most likely model. For LL, the largest score is the most likely model. The type indicates which parameters are estimated. This function allows all parameters (`type = 'free'`), only alpha (`type = 'fixed'`), only alpha and variance (`type = 'fixed_2'`), and only variance (`type ='fixed_3`) to be estimated for each mixture.

Examples

```
out <- quackBeta(xm[1:100,], samplename = "sample1", cores = 1)
```

---

quackBetaBinom *Model Selection - Expectation Maximization - Beta-Binomial Mixture*

Description

This function uses the expectation maximization of both the beta-binomial and beta-binomial-uniform mixture models for model selection. Here we can run up to 32 mixture models.

Usage

```
quackBetaBinom(
  xm,
  samplename,
  cores,
  parallel = FALSE,
  trunc = c(0, 0),
  lowvar = FALSE,
  tau = NA,
  error = NA,
  free = FALSE,
  verbose = TRUE
)
```

Arguments

xm	Matrix with two columns with total coverage and coverage for a randomly sampled allele.
samplename	Name of sample to be included in output.
cores	Threads available to run process in parallel.
parallel	default = FALSE, set to true if cores > 1.
trunc	List of two values representing the lower and upper bounds for allele frequency truncation , c_L and c_U. If allele frequency truncation was done to remove error, then you do not need to truncate the expected. If no truncation has been done, this should be set to c(0,0), which is the default.
lowvar	Default to FALSE. When false, variance is equal to 0.01. If set to TRUE and tau and error are not provided, the variance will be set as 0.001.
tau	Sequencing overdispersion parameter. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
error	Sequencing error rate. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
free	default = FALSE, skip the free model calculation and does not calculate delta log-likelihood.
verbose	Default to TRUE. If TRUE, progress messages will be printed to the console.

Value

BIC scores and log-likelihood (LL) mixture models including diploid, triploid, tetraploid, pentaploid, and hexaploid. When free = TRUE, the delta log-likelihood (dLL) is calculated based on the associated free model (without or with a uniform mixture). For BIC or delta-log likelihood, the smallest score is the most likely model. For LL, the largest score is the most likely model. The type indicates which parameters are estimated. This function allows all parameters (`type = 'free'`), only alpha (`type = 'fixed'`), only alpha and variance (`type = 'fixed_2'`), and only variance (`type ='fixed_3`) to be estimated for each mixture.

Examples

```
if(exists("crazy")){
  out <- quackBetaBinom(xm[1:100,], samplename = "sample1", cores = 1)
}
```

---

quackit *Model Selection - Based on BIC or Log-Likelihood*

Description

This function is for model interpretation.

Usage

```
quackit(
  model_out,
  summary_statistic = "BIC",
  mixtures = c("diploid", "triploid", "tetraploid", "hexaploid", "pentaploid")
)
```

Arguments

model_out	Data frame containing, at minimum, columns labeled LL, type, mixture, distribution, and BIC.
summary_statistic	May be equal to BIC or LL.
mixtures	Defaults to c("diploid", "triploid", "tetraploid", "hexaploid", "pentaploid").

Value

Returns data frame with the most likely model for each set of mixtures. Includes the best and second best mixtures, as well as the difference between the two. We only use BIC or LL to compare within each distribution and type. To identify the most accurate model, you will need to compare accuracy across distributions and types using a set of known samples. The distributions include Normal, Beta, and Beta-Binomial - each with and without a uniform mixture. The type indicates which parameters are estimated for the mixtures: all parameters (type = 'free', only used to calculate delta log-likelihood), only alpha (type = 'fixed'), only alpha and variance (type = 'fixed_2'), and only variance (type ='fixed_3) to be estimated for each mixture.

Examples

```
out <- quackNormal(xm[1:100,], samplename = "sample1", cores = 1)
goose <- quackit(out)
```

quackNboots *Bootstrapping - Expectation Maximization - Optimal Distribution and Type*

Description

This function was made to assist with bootstrap replication for a set of models run a subset of models based on a selected distribution and type. There are many limitations to this function to make this tractable, as there are 128 models that could be run with our package. Here we do not include models or comparisons we found unhelpful, this includes the nQuire implementation and log-likelihood ratio tests.

Usage

```
quackNboots(
  xm,
  nboots = 100,
  distribution,
  type,
  uniform,
  mixtures = c("diploid", "triploid", "tetraploid", "hexaploid", "pentaploid"),
  samplename,
  trunc = c(0, 0),
  lowvar = FALSE,
  tau = NA,
  error = NA
)
```

Arguments

xm	Matrix with two columns with total coverage and coverage for a randomly sampled allele.
nboots	Number of bootstrap replicates to examine.
distribution	May be set to normal, beta, or beta-binomial. We do not include the implementation with nQuire.
type	May be equal to fixed, fixed_2, or fixed_3.
uniform	If equal to 1, a uniform mixture is included. If equal to 0, no uniform mixture is included.
mixtures	Defaults to `c("diploid", "triploid", "tetraploid", "hexaploid", "pentaploid")`.
samplename	Name of sample to be included in output.
trunc	List of two values representing the lower and upper bounds for allele frequency truncation ,c_L and c_U. If allele frequency truncation was done to remove error, then you do not need to truncate the expected. If no truncation has been done, this should be set to c(0,0), which is the default.

lowvar	Default to FALSE. When false, variance is equal to 0.01. If set to TRUE and tau and error are not provided, the variance will be set as 0.001.
tau	Sequencing overdispersion parameter. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
error	Sequencing error rate. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.

Value

BIC scores and log-likelihood (LL) for included mixture models. For both, the smallest score is the most likely model.

@export

Examples

```
out <- quackNboots(xm[1:100,],
                   distribution = "normal",
                   type = "fixed",
                   uniform = 1,
                   samplename = "sample1",
                   nboots = 2)
```

---

quackNormal *Model Selection - Expectation Maximization - Normal Mixture*

Description

This function uses the expectation maximization of both the normal and normal-uniform mixture models for model selection. Here we can run up to 32 mixture models.

Usage

```
quackNormal(
  xm,
  samplename,
  cores,
  parallel = FALSE,
  trunc = c(0, 0),
  lowvar = FALSE,
  tau = NA,
  error = NA,
  free = FALSE,
  verbose = TRUE
)
```

Arguments

xm	Matrix with two columns with total coverage and coverage for a randomly sampled allele.
samplename	Name of sample to be included in output.
cores	Threads available to run process in parallel.
parallel	default = FALSE, set to true if cores > 1.
trunc	List of two values representing the lower and upper bounds for allele frequency truncation, c_L and c_U. If allele frequency truncation was done to remove error, then you do not need to truncate the expected. If no truncation has been done, this should be set to c(0,0), which is the default.
lowvar	Default to FALSE. When false, variance is equal to 0.01. If set to TRUE and tau and error are not provided, the variance will be set as 0.001.
tau	Sequencing overdispersion parameter. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
error	Sequencing error rate. If tau and error are provided, the variance of each mixture will be inferred from these values.
free	default = FALSE, skip the free model calculation and does not calculate delta log-likelihood.
verbose	Default to TRUE. If TRUE, progress messages will be printed to the console.

Value

BIC scores and log-likelihood (LL) mixture models including diploid, triploid, tetraploid, pentaploid, and hexaploid. When free = TRUE, the delta log-likelihood (dLL) is calculated based on the associated free model (without or with a uniform mixture). For BIC or delta-log likelihood, the smallest score is the most likely model. For LL, the largest score is the most likely model. The type indicates which parameters are estimated. This function allows all parameters (`type = 'free'`), only alpha (`type = 'fixed'`), only alpha and variance (`type = 'fixed_2'`), and only variance (`type ='fixed_3`) to be estimated for each mixture.

Examples

```
out <- quackNormalNQ(xm[1:100,], samplename = "sample1", cores = 1)
```

---

quackNormalNQ *Model Selection - Expectation Maximization - Normal Mixture (nQuire)*

Description

This function uses the expectation maximization of both the normal and normal-uniform mixture models for model selection based on the nQuire approach. Here we can run up to 32 mixture models.

Usage

```
quackNormalNQ(
  xm,
  samplename,
  cores,
  parallel = FALSE,
  trunc = c(0, 0),
  lowvar = FALSE,
  tau = NA,
  error = NA,
  free = FALSE,
  verbose = TRUE
)
```

Arguments

xm	Matrix with two columns with total coverage and coverage for a randomly sampled allele.
samplename	Name of sample to be included in output.
cores	Threads available to run process in parallel.
parallel	default = FALSE, set to true if cores > 1.
trunc	List of two values representing the lower and upper bounds for allele frequency truncation , c_L and c_U. If allele frequency truncation was done to remove error, then you do not need to truncate the expected. If no truncation has been done, this should be set to c(0,0), which is the default.
lowvar	Default to FALSE. When false, variance is equal to 0.01. If set to TRUE and tau and error are not provided, the variance will be set as 0.001.
tau	Sequencing overdispersion parameter. If tau and error are provided, the variance of each mixture will be inferred from these values. If not, the variance by default is equal to 0.01 or 0.001.
error	Sequencing error rate. If tau and error are provided, the variance of each mixture will be inferred from these values.
free	default = FALSE, skip the free model calculation and does not calculate delta log-likelihood.
verbose	Default to TRUE. If TRUE, progress messages will be printed to the console.

Value

BIC scores and log-likelihood (LL) mixture models including diploid, triploid, tetraploid, pentaploid, and hexaploid. When free = TRUE, the delta log-likelihood (dLL) is calculated based on the associated free model (without or with a uniform mixture). For BIC or delta-log likelihood, the smallest score is the most likely model. For LL, the largest score is the most likely model. The type indicates which parameters are estimated. This function allows all parameters (`type = 'free'`), only alpha (`type = 'fixed'`), only alpha and variance (`type = 'fixed_2'`), and only variance (`type ='fixed_3`) to be estimated for each mixture.

Examples

```
out <- quackNormalNQ(xm[1:100,], samplename = "sample1", cores = 1)
```

`resample_xm` *Calculate Alpha and Beta from Mean and Variance*

Description

Calculate Alpha and Beta from Mean and Variance

Usage

```
resample_xm(xm, n)
```

Arguments

`xm`	Matrix with total coverage and coverage at a randomly sampled allele.
`n`	Length of matrix.

Value

Randomly sampled matrix.

Examples

```
outdf <- resample_xm(as.matrix(xm), n = 10)
```

`setconvert` *Calculate Variance from Mean, Tau, and Sequencing Error*

Description

This function is used to replace variance in mixture model sets.

Usage

```
setconvert(set, tau, error)
```

Arguments

`set`	A list of lists, each of the lists must contain avec, mvec, and svec.
`tau`	Sequence overdispersion parameter for read counts.
`error`	Sequencing error rate.

Value

Mean and variance for the associated tau and error.

Examples

```
set <- c()
set[[1]] =  list(avec = c(1.00), mvec = c(0.50), svec = c(0.01));
set[[2]] =  list(avec = c(0.50, 0.50), mvec = c(0.67, 0.33), svec = c(0.01, 0.01));
exset <- setconvert(set, tau = 0.01, error = 0.001)
```

SetupBasicExample *Setup Basic Example*

Description

This function was made to download all files needed to run the Basic Example vignette. It downloads a zipped file from Zenodo and unzips it to the "inst/extdata/" directory within the nQuack package.

Usage

```
SetupBasicExample(overwrite = FALSE)
```

Arguments

`overwrite`	Logical. If TRUE, the function will overwrite any existing files in the data directory. Default is FALSE.

Value

This function downloads files to the "inst/extdata/" directory within the nQuack package and does not return any value. The function will print messages as it downloads to keep you updated on the progress.

Examples

```
if(exists("crazy")){
   SetupBasicExample(overwrite = FALSE)
}
```

`sim.ind.BB` *Simulate Allele Counts for Single Individual - Beta-Binomial Distribution*

Description

This function is used to simulate coverage of each allele at biallelic heterozygous sites assuming a beta binomial distribution. Here we sample sequence depth from a truncated poisson distribution between a set minimum, maximum, and lambda. Only heterozygous sites are returned. Based on input variables, the sites may be filtered based on the total coverage (`filter.coverage`), allele sequencing coverage (`filter.error`), or allele frequency (`filter.freq`).

Usage

```
sim.ind.BB(
  mvec,
  avec,
  svec,
  error = 0.001,
  s.size = 50000,
  lambda = 11,
  max.coverage = 20,
  min.coverage = 2,
  filter.coverage = TRUE,
  max.depth.quantile.prob = 0.9,
  filter.error = TRUE,
  filter.freq = FALSE,
  trunc = c(0, 0),
  sampled = TRUE
)
```

Arguments

`mvec`	Vector of mean values of allele frequency.
`avec`	Vector of alpha values representing the proportion expected of each mean.
`svec`	Vector of variance values.
`error`	Sequencing error rate. Default as 0.001, or very low error.
`s.size`	Number of biallelic sites to generate. Defaults to 50000. Warning, the number of sites generated will not be the number of sites returned due to filtering steps.
`lambda`	Set lambda for the truncated poisson distrubtion. Defaults to 11.
`max.coverage`	Maximum sequencing depth per site. Defaults to 20.
`min.coverage`	Minimum sequencing depth per site. Defaults to 2.
`filter.coverage`	Default as TRUE. Filters to only retain sites where total sequencing depth is greater than the provided minimum coverage and less than the max quantile depth (set with the max.depth.quantile.prob).

`max.depth.quantile.prob`
: Maximum depth quantile probability. Defaults to 0.9.

`filter.error`
: Default as TRUE. Filter to only retain sites where allele coverage is greater than the sequencing error rate times the total coverage, but less than one minus the sequencing error rate times the total coverage.

`filter.freq`
: Default as FALSE. When set to true, sites are filtered based on provided `trunc`.

`trunc`
: List of two values representing the lower and upper bounds,c_L and c_U. Defaults as c(0,0) to represent no truncation.

`sampled`
: Default as TRUE. Will randomly sample allele A or allele B, then return a data frame with total coverage and coverage of a randomly sampled allele will be returned.

Value

If sampled = FALSE, a data frame with total coverage, coverage of allele A, and coverage of allele B will be returned. If sampled = TRUE, a data frame with total coverage and coverage of a randomly sampled allele will be returned.

Examples

```
xm <- sim.ind.BB(mvec = c(0.5), avec = c(1), svec=c(0.01), s.size = 100)
```

---

sim.ind.BB.tau *Simulate Allele Counts for Single Individual - Beta-Binomial Distribution with Overdispersion and Error*

Description

This function is used to simulate the frequency of biallelic heterozygous sites assuming a beta-binomial distribution. Here we sample sequence depth from a truncated poisson distribution between a set minimum, maximum, and lambda. Only heterozygous sites are returned. Based on input variables, the sites may be filtered based on the total coverage (`filter.coverage`), allele sequencing coverage (`filter.error`), or allele frequency (`filter.freq`).

Usage

```
sim.ind.BB.tau(
  mvec,
  avec,
  tau = 0.01,
  error = 0.001,
  s.size = 50000,
  lambda = 11,
  max.coverage = 20,
  min.coverage = 2,
  filter.coverage = TRUE,
```

```
  max.depth.quantile.prob = 0.9,
  filter.error = TRUE,
  filter.freq = FALSE,
  trunc = c(0, 0),
  sampled = TRUE
)
```

Arguments

mvec	Vector of mean values of allele frequency.
avec	Vector of alpha values representing the proportion expected of each mean.
tau	Overdispersion parameter. Defaults to 0.01.
error	Sequencing error rate. Defaults to 0.001.
s.size	Number of biallelic sites to generate. Defaults to 50000. Warning, the number of sites generated will not be the number of sites returned due to filtering steps.
lambda	Set lambda for the truncated poisson distrubtion. Defaults to 11.
max.coverage	Maximum sequencing depth per site. Defaults to 20.
min.coverage	Minimum sequencing depth per site. Defaults to 2.
filter.coverage	Default as TRUE. Filters to only retain sites where total sequencing depth is greater than the provided minimum coverage and less than the max quantile depth (set with the max.depth.quantile.prob).
max.depth.quantile.prob	Maximum depth quantile probability. Defaults to 0.9.
filter.error	Default as TRUE. Filter to only retain sites where allele coverage is greater than the sequencing error rate times the total coverage, but less than one minus the sequencing error rate times the total coverage.
filter.freq	Default as FALSE. When set to true, sites are filtered based on provided trunc.
trunc	List of two values representing the lower and upper bounds, c_L and c_U. Defaults as c(0,0) to represent no truncation.
sampled	Default as TRUE. Will randomly sample allele A or allele B, then return a data frame with total coverage and coverage of a randomly sampled allele will be returned.

Value

If sampled = FALSE, a data frame with total coverage, coverage of allele A, and coverage of allele B will be returned. If sampled = TRUE, a data frame with total coverage and coverage of a randomly sampled allele will be returned.

Examples

```
xm <- sim.ind.BB.tau(mvec = c(0.5), avec = c(1), s.size = 100)
```

`sim.ind.simple` *Simulate Allele Counts for Single Individual - Simple Approach*

Description

This function is used to simulate coverage of each allele at biallelic heterozygous sites assuming a binomial distribution.

Usage

```
sim.ind.simple(mvec, cover = 100, s.size = 50000, sampled = TRUE)
```

Arguments

`mvec`	Vector of means.
`cover`	Coverage of sites.
`s.size`	Number of biallelic sites to generate. Defaults to 50000. Warning, the number of sites generated will not be the number of sites returned due to filtering steps.
`sampled`	Default as TRUE. Will randomly sample allele A or allele B, then return a data frame with total coverage and coverage of a randomly sampled allele will be returned.

Value

If sampled = FALSE, a data frame with total coverage, coverage of allele A, and coverage of allele B will be returned. If sampled = TRUE, a data frame with total coverage and coverage of a randomly sampled allele will be returned.

Examples

```
xm <- sim.ind.simple(mvec = c(0.5), s.size = 100)
```

Index

alphabetacalc, 2
alphabetacalctau, 3
alphabetacalctauvec, 4
alphabetacalcvec, 4

Bclean, 5
bestquack, 6

denoise_data, 7

emstepB, 8
emstepB3, 9
emstepBB, 10
emstepBBU, 11
emstepBU, 12
emstepN, 13
emstepNA, 14
emstepNU, 15
emstepNUA, 16
estepB3, 17

muvarcalcvec, 17

nQuire_reformat, 18

prepare_data, 19
process_data, 20
process_nquire, 21
process_rcpp, 21

quackBeta, 22
quackBetaBinom, 23
quackit, 25
quackNboots, 26
quackNormal, 27
quackNormalNQ, 28

resample_xm, 30

setconvert, 30
SetupBasicExample, 31
sim.ind.BB, 32
sim.ind.BB.tau, 33
sim.ind.simple, 35