Package ‘segtest’

January 16, 2025

Title Tests for Segregation Distortion in Polyploids

Version 1.0.2

Description Provides a suite of tests for
segregation distortion in F1 polyploid populations (for
now, just tetraploids). This is under different assumptions of
meiosis. Details of these methods are described in
Gerard et al. (2025) <doi:10.1007/s00122-025-04816-z>.
This material is based upon work supported by the
National Science Foundation under Grant No. 2132247. The
opinions, findings, and conclusions or recommendations expressed
are those of the author and do not necessarily reflect the views
of the National Science Foundation.

License GPL (>= 3)

BugReports `https://github.com/dcgerard/segtest/issues`

URL `https://github.com/dcgerard/segtest`

Encoding UTF-8

RoxygenNote 7.3.2

Biarch true

Depends R (>= 3.4.0)

Imports doFuture, doRNG, foreach, future, iterators, Rcpp, updog

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

Config/testthat/edition 3

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

LazyData true

NeedsCompilation yes

Author David Gerard [aut, cre] (<`https://orcid.org/0000-0001-9450-5023`>),
Mira Thakkar [aut],
NSF DBI 2132247 [fnd]
(https://www.nsf.gov/awardsearch/showAward?AWD_ID=2132247)

Maintainer David Gerard <gerard.1787@gmail.com>

Repository CRAN

Date/Publication 2025-01-16 17:10:02 UTC

Contents

`chisq_g4` *Chi Square test when genotypes are known*

Description

This chi-squared test is run under the assumption of no double reduction and no preferential pairing.

Usage

```
chisq_g4(x, g1, g2)
```

Arguments

`x`	Vector of observed genotype counts
`g1`	Parent 1's genotype
`g2`	Parent 2's genotype

Value

A list containing the chi-squared statistic, degrees of freedom, and p-value.

Author(s)

Mira Thakkar and David Gerard

Examples

```
x <- c(1, 2, 4, 3, 0)
g1 <- 2
g2 <- 2
chisq_g4(x, g1, g2)

x <- c(10, 25, 10, 0, 0)
g1 <- 1
g2 <- 1
chisq_g4(x, g1, g2)
```

chisq_gl4 *Chi-Sq for GL*

Description

Calculates the MLE genotype and runs a chi-squared test assuming no double reduction and no preferential pairing.

Usage

```
chisq_gl4(gl, g1, g2)
```

Arguments

`gl`	A matrix of offspring genotype log-likelihoods. The rows index the individuals and the columns index the possible genotypes. So `gl[i, k]` is the offspring genotype log-likelihood for individual i and genotype k-1.
`g1`	The first parent's genotype.
`g2`	The second parent's genotype.

Value

A list containing the chi-squared statistic, degrees of freedom, and p-value.

Author(s)

Mira Thakkar and David Gerard

Examples

```
## null sim
set.seed(1)
g1 <- 2
g2 <- 2
gl <- simf1gl(n = 25, g1 = g1, g2 = g2, alpha = 0, xi2 = 1/3)
chisq_gl4(gl = gl, g1 = g1, g2 = g2)
```

em_li *EM algorithm from Li (2011)*

Description

EM algorithm to estimate prior genotype probabilities from genotype likelihoods.

Usage

```
em_li(B, itermax = 100L, eps = 1e-05)
```

Arguments

`B`	Matrix of genotype log-likelihoods. The rows index the individuals and the columns index the genotypes.
`itermax`	The maximum number of iterations.
`eps`	The stopping criteria.

Value

A vector of log prior probabilities for each genotype.

Author(s)

David Gerard

References

- Li, H. (2011). A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, 27(21), 2987-2993. doi:10.1093/bioinformatics/btr509

Examples

```
# Simulate some data
set.seed(1)
gl <- simgl(nvec = c(3, 2, 4, 1, 2))
# Run em
lprob <- em_li(B = gl)
# Exponentiate to get probabilities
prob <- exp(c(lprob))
prob
```

gcount_to_gvec *Converts genotype counts to genotype vectors.*

Description

Converts genotype counts to genotype vectors.

Usage

```
gcount_to_gvec(gcount)
```

Arguments

gcount	The vector of genotype counts.

Value

A vector of length `sum(gcount)`, containing `gcount[1]` copies of `0`, `gcount[2]` copies of `1`, `gcount[3]` copies of `2`, etc.

Author(s)

David Gerard

See Also

`gvec_to_gcount()`

Examples

```
gcount <- c(1, 2, 3, 0, 5)
gcount_to_gvec(gcount = gcount)
```

gvec_to_gcount *Inverse function of* `gcount_to_gvec()`.

Description

Inverse function of `gcount_to_gvec()`.

Usage

```
gvec_to_gcount(gvec, ploidy = 4)
```

Arguments

`gvec`	The vector of genotypes. `gvec[i]` is the genotype for individual i.
`ploidy`	The ploidy of the species.

Value

A vector of counts. Element `k` is the number of individuals with genotype `k-1`.

Author(s)

David Gerard

See Also

`gcount_to_gvec()`

Examples

```
gvec <- c(1, 2, 3, 2, 3, 1, 4, 0, 1, 0, 0, 1, 0, 0)
gvec_to_gcount(gvec = gvec)
```

`is_valid_2` *Tests if the two parameter model is valid*

Description

There is a dependence on the bounds of two-parameter model. This function returns TRUE if those bounds are satisfied and FALSE otherwise.

Usage

```
is_valid_2(dr, pp, drbound = 1/6)
```

Arguments

`dr`	The double reduction rate.
`pp`	The preferential pairing parameter.
`drbound`	The maximum double reduction rate possible.

Value

`TRUE` if the model is valid, `FALSE` otherwise.

Author(s)

David Gerard

Examples

```
TOL <- 1e-6
is_valid_2(dr = 1/6, pp = 1/3, drbound = 1/6) # Valid
is_valid_2(dr = 1/6, pp = 1/3 - TOL, drbound = 1/6) # Not valid
is_valid_2(dr = 1/6, pp = 1/3 + TOL, drbound = 1/6) # Not valid
```

iter.array *Iterator over array*

Description

Iterator over array

Usage

```
## S3 method for class 'array'
iter(obj, by = 1, recycle = FALSE, ...)
```

Arguments

`obj`	An array.
`by`	The dimension to iterate over.
`recycle`	Should the iterator reset?
`...`	not used

Value

An iterator. This is an S3 arrayiter object, used in conjunction with nextElem to iterate over one index of an array.

Author(s)

David Gerard

See Also

`nextElem.arrayiter()`

Examples

```
glist <- multidog_to_g(mout = ufit, type = "all_gl", p1 = "indigocrisp", p2 = "sweetcrisp")
g <- iterators::iter(glist$g, by = 3)
head(iterators::nextElem(g))
head(iterators::nextElem(g))
head(iterators::nextElem(g))
```

`like_gknown_2` *Likelihood under three parameter model when genotypes are known*

Description

This is under the two parameter model.

Usage

```
like_gknown_2(x, alpha, xi1, xi2, g1, g2, log_p = TRUE, pen = 0)
```

Arguments

`x`	A vector of length 5. `x[i]` is the count of individuals with genotype `i-1`.
`alpha`	The double reduction rate.
`xi1`	The preferential pairing parameter of parent 1.
`xi2`	The preferential pairing parameter of parent 2.
`g1`	Parent 1's genotype.
`g2`	Parent 2's genotype.
`log_p`	A logical. Should we return the log likelihood or not?
`pen`	A tiny penalty to help with numerical stability

Value

The (log) likelihood.

Author(s)

David Gerard

Examples

```
x <- c(1, 4, 5, 3, 1)
alpha <- 0.01
xi1 <- 0.5
xi2 <- 0.3
g1 <- 1
g2 <- 2
like_gknown_2(
  x = x,
  alpha = alpha,
  xi1 = xi1,
  xi2 = xi2,
  g1 = g1,
  g2 = g2)
```

`like_gknown_3` *Likelihood under three parameter model when genotypes are known*

Description

This is under the three parameter model.

Usage

```
like_gknown_3(x, tau, beta, gamma1, gamma2, g1, g2, log_p = TRUE, pen = 0)
```

Arguments

x	A vector of length 5. `x[i]` is the count of individuals with genotype `i-1`.
tau	The probability of quadrivalent formation.
beta	The probability of double reduction given quadrivalent formation.
gamma1	The probability of AA_aa pairing for parent 1.
gamma2	The probability of AA_aa pairing for parent 2.
g1	Parent 1's genotype.
g2	Parent 2's genotype.
log_p	A logical. Should we return the log likelihood or not?
pen	A tiny penalty to help with numerical stability

Value

The (log) likelihood.

Author(s)

David Gerard

Examples

```
x <- c(1, 4, 5, 3, 1)
tau <- 0.5
beta <- 0.1
gamma1 <- 0.5
gamma2 <- 0.3
g1 <- 1
g2 <- 2
like_gknown_3(
  x = x,
  tau = tau,
  beta = beta,
  gamma1 = gamma1,
  gamma2 = gamma2,
  g1 = g1,
  g2 = g2)
```

`like_glpknown_2` *Likelihood under three parameter model when using offspring genotypes likelihoods but parent genotypes are known.*

Description

This is under the two parameter model.

Usage

```
like_glpknown_2(gl, alpha, xi1, xi2, g1, g2, log_p = TRUE)
```

Arguments

`gl`	The matrix of genotype likelihoods of the offspring. Rows index The individuals, columns index the genotypes.
`alpha`	The double reduction rate.
`xi1`	The preferential pairing parameter of parent 1.
`xi2`	The preferential pairing parameter of parent 2.
`g1`	Parent 1's genotype.
`g2`	Parent 2's genotype.
`log_p`	A logical. Should we return the log likelihood or not?

Value

The (log) likelihood of the two parameter model when using genotype likelihoods.

Author(s)

David Gerard

Examples

```
g1 <- 1
g2 <- 0
gl <- simf1gl(
  n = 25,
  g1 = g1,
  g2 = g2,
  rd = 10,
  alpha = 0,
  xi1 = 1/3,
  xi2 = 1/3)
```

```
like_glpknown_2(
  gl = gl,
  alpha = 0.01,
  xi1 = 0.5,
  xi2 = 0.3,
  g1 = g1,
  g2 = g2,
  log_p = TRUE)
```

like_glpknown_3 *Likelihood under three parameter model when using offspring genotypes likelihoods but parent genotypes are known.*

Description

This is under the three parameter model.

Usage

```
like_glpknown_3(gl, tau, beta, gamma1, gamma2, g1, g2, log_p = TRUE)
```

Arguments

gl	The matrix of genotype likelihoods of the offspring. Rows index The individuals, columns index the genotypes.
tau	The probability of quadrivalent formation.
beta	The probability of double reduction given quadrivalent formation.
gamma1	The probability of AA_aa pairing for parent 1.
gamma2	The probability of AA_aa pairing for parent 2.
g1	Parent 1's genotype.
g2	Parent 2's genotype.
log_p	A logical. Should we return the log likelihood or not?

Value

The (log) likelihood of the three parameter model when using genotype likelihoods.

Author(s)

David Gerard

Examples

```
g1 <- 1
g2 <- 0
gl <- simf1gl(
  n = 25,
  g1 = g1,
  g2 = g2,
  rd = 10,
  alpha = 0,
  xi1 = 1/3,
  xi2 = 1/3)
like_glpknown_3(
  gl = gl,
  tau = 1/2,
  beta = 1/12,
  gamma1 = 1/3,
  gamma2 = 1/3,
  g1 = g1,
  g2 = g2,
  log_p = TRUE)
```

`llike_li` *Objective function for* `em_li()`

Description

Objective function for `em_li()`

Usage

```
llike_li(B, lpivec)
```

Arguments

`B`	The log-likelihood matrix. Rows are individuals columns are genotypes.
`lpivec`	The log prior vector.

Value

The log-likelihood of a vector of genotype frequencies when using genotype likelihoods. This is from Li (2011).

Author(s)

David Gerard

References

- Li, H. (2011). A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, 27(21), 2987-2993. doi:10.1093/bioinformatics/btr509

Examples

```
# Simulate some data
set.seed(1)
gl <- simgl(nvec = c(3, 2, 4, 1, 2))
# Log-likelihood at given log-priors
prob <- c(0.1, 0.2, 0.4, 0.2, 0.1)
lprob <- log(prob)
llike_li(B = gl, lpivec = lprob)
```

`lrt_men_g4` *Likelihood ratio test for segregation distortion with known genotypes*

Description

This will run a likelihood ratio test using the genotypes of an F1 population of tetraploids for the null of Mendelian segregation (accounting for double reduction and preferential pairing) against the alternative of segregation distortion. This is when the genotypes are assumed known.

Usage

```
lrt_men_g4(
  x,
  g1,
  g2,
  drbound = 1/6,
  pp = TRUE,
  dr = TRUE,
  alpha = 0,
  xi1 = 1/3,
  xi2 = 1/3
)
```

Arguments

`x`	A vector of genotype counts. `x[i]` is the number of offspring with genotype `i-1`.
`g1`	The genotype of parent 1.
`g2`	The genotype of parent 2.
`drbound`	The maximum rate of double reduction. A default of 1/6 is provided, which is the rate under the complete equational segregation model of meiosis.

`pp`	A logical. Should we account for preferential pairing (`TRUE`) or not (`FALSE`)?
`dr`	A logical. Should we account for double reduction (`TRUE`) or not (`FALSE`)?
`alpha`	If `dr = FALSE`, this is the known rate of double reduction.
`xi1`	If `pp = FALSE`, this is the known preferential pairing parameter of parent 1.
`xi2`	If `pp = FALSE`, this is the known preferential pairing parameter of parent 2.

Value

A list with the following elements

`statistic` The log-likelihood ratio test statistic.

`df` The degrees of freedom.

`p_value` The p-value.

`alpha` The estimated double reduction rate.

`xi1` The estimated preferential pairing parameter of parent 1.

`xi2` The estimated preferential pairing parameter of parent 2.

Impossible genotypes

Some offspring genotype combinations are impossible given the parental genotypes. If these impossible genotypes combinations show up, we return a p-value of 0, a log-likelihood ratio statistic of Infinity, and missing values for all other return items. The impossible genotypes are:

`g1 = 0 && g2 = 0` Only offspring genotypes of 0 are possible.

`g1 = 4 && g2 = 4` Only offspring genotypes of 4 are possible.

`g1 = 0 && g2 = 4 || g1 == 4 && g2 == 0` Only offspring genotypes of 2 are possible.

`g1 = 0 && g2 %in% c(1, 2, 3) || g1 = %in% c(1, 2, 3) && g2 == 0` Only offspring genotypes of 0, 1, and 2 are possible.

`g1 = 4 && g2 %in% c(1, 2, 3) || g1 = %in% c(1, 2, 3) && g2 == 4` Only offspring genotypes of 2, 3, and 4 are possible.

Unidentified parameters

When `g1 = 2` or `g2 = 2` (or both), the model is not identified and those estimates (`alpha`, `xi1`, and `xi2`) are meaningless. Do NOT interpret them.

The estimate of `alpha` (double reduction rate) IS identified as long as at least one parent is simplex, and no parent is duplex. However, the estimates of the double reduction rate have extremely high variance.

Author(s)

David Gerard

Examples

```
set.seed(100)
gf <- offspring_gf_2(alpha = 1/12, xi1 = 0.2, xi2 = 0.6, p1 = 1, p2 = 0)
x <- offspring_geno(gf = gf, n = 100)
lrt_men_g4(x = x, g1 = 1, g2 = 0)
```

`lrt_men_gl4` *Likelihood ratio test using genotype likelihoods.*

Description

This will run a likelihood ratio test using the genotypes of an F1 population of tetraploids for the null of Mendelian segregation (accounting for double reduction and preferential pairing) against the alternative of segregation distortion. This is when genotype uncertainty is accounted for through genotype likelihoods.

Usage

```
lrt_men_gl4(
  gl,
  g1 = NULL,
  g2 = NULL,
  drbound = 1/6,
  pp = TRUE,
  dr = TRUE,
  alpha = 0,
  xi1 = 1/3,
  xi2 = 1/3
)
```

Arguments

`gl`	The genotype log-likelihoods. The rows index the individuals and the columns index the genotypes.
`g1`	Either parent 1's genotype, or parent 1's genotype log-likelihoods.
`g2`	Either parent 2's genotype, or parent 2's genotype log-likelihoods.
`drbound`	The upper bound on the double reduction rate.
`pp`	Is (partial) preferential pairing possible (`TRUE`) or not (`FALSE`)?
`dr`	Is double reduction possible (`TRUE`) or not (`FALSE`)?
`alpha`	If `dr = FALSE`, this is the known rate of double reduction.
`xi1`	If `pp = FALSE`, this is the known preferential pairing parameter of parent 1.
`xi2`	If `pp = FALSE`, this is the known preferential pairing parameter of parent 2.

Value

A list with the following elements

- `statistic` The log-likelihood ratio test statistic.
- `df` The degrees of freedom.
- `p_value` The p-value.
- `alpha` The estimated double reduction rate.
- `xi1` The estimated preferential pairing parameter of parent 1.
- `xi2` The estimated preferential pairing parameter of parent 2.

Unidentified parameters

When `g1 = 2` or `g2 = 2` (or both), the model is not identified and those estimates (`alpha`, `xi1`, and `xi2`) are meaningless. Do NOT interpret them.

The estimate of `alpha` (double reduction rate) IS identified as long as at least one parent is simplex, and no parent is duplex. However, the estimates of the double reduction rate have extremely high variance.

Author(s)

David Gerard

Examples

```
## null simulation
set.seed(1)
g1 <- 2
g2 <- 2
gl <- simf1gl(n = 25, g1 = g1, g2 = g2, alpha = 1/12, xi2 = 1/2)

## LRT when parent genotypes are known.
lrt_men_gl4(gl = gl, g1 = g1, g2 = g2)

## LRT when parent genotypes are not known
lrt_men_gl4(gl = gl)

## Alternative simulation
gl <- simgl(nvec = rep(5, 5))
lrt_men_gl4(gl = gl, g1 = g1, g2 = g2)
```

multidog_to_g — *Converts multidog output to a format usable for multi_lrt()*

Description

Converts multidog output to a format usable for multi_lrt()

Usage

```
multidog_to_g(
  mout,
  type = c("off_gl", "all_gl", "all_g", "off_g"),
  p1 = NULL,
  p2 = NULL,
  ploidy = 4
)
```

Arguments

`mout`	The output of `multidog()`.
`type`	`"off_gl"` Genotype likelihoods of offspring but not parents. This is the typical choice if you used the "f1" or "f1pp" options when genotyping. `"all_gl"` Genotype likelihoods of offspring and parents. This is only done if you did *not* use the "f1" or "f1pp" options when genotyping. If this is the case, then you need to specify which individuals are the parents. `"off_g"` Genotypes, assuming that they are known. You used the "f1" or "f1pp" option when genotyping. `"all_g"` Genotypes, assuming that they are known. You did *not* use the "f1" or "f1pp" option when genotyping. If this is the case, then you need to specify which individuals are the parents.
`p1`	The first parent name if using `type = "all_gl"` or `type = "all_g"`.
`p2`	The second parent name if using `type = "all_gl"` or `type = "all_g"`.
`ploidy`	The ploidy. Note that most methods in this package (including those in `multi_lrt()`) assume that the ploidy is 4. But we allow for arbitrary ploidy in this function since it might be useful in the future.

Value

A list with the following elements

g Either a matrix of counts, where the columns index the genotype and the rows index the loci (`type = "all_g"` or `type = "off_g"`). Or an array of genotype (natural) log-likelihoods where the rows index the loci, the columns index the individuals, and the slices index the genotypes (`type = "all_gl"` or `type = "off_gl"`).

p1 Either a vector of known parental genotypes (type = "off_gl", type = "all_g" or type = "off_g"). Or a matrix of genotype (natural) log-likelihoods where the rows index the loci and the columns index the genotypes (type = "all_gl").

p2 Either a vector of known parental genotypes (type = "off_gl", type = "all_g" or type = "off_g"). Or a matrix of genotype (natural) log-likelihoods where the rows index the loci and the columns index the genotypes (type = "all_gl").

Author(s)

David Gerard

Examples

```
multidog_to_g(mout = ufit, type = "all_g", p1 = "indigocrisp", p2 = "sweetcrisp")
multidog_to_g(mout = ufit, type = "all_gl", p1 = "indigocrisp", p2 = "sweetcrisp")
multidog_to_g(mout = ufit2, type = "off_g")
multidog_to_g(mout = ufit2, type = "off_gl")
multidog_to_g(mout = ufit3, type = "off_g")
multidog_to_g(mout = ufit3, type = "off_gl")
```

multi_lrt *Parallelized likelihood ratio test for segregation distortion.*

Description

Uses the future package to implement parallelization support for the likelihood ratio tests for segregation distortion. Right now, this is only supported for tetraploids (allo, auto, or segmental). This function is only somewhat tested (the single-locus LRT functions in the "See Also" section are very well tested). So please send any bugs you notice to https://github.com/dcgerard/segtest/issues.

Usage

```
multi_lrt(
  g,
  p1,
  p2,
  drbound = 1/6,
  pp = TRUE,
  dr = TRUE,
  alpha = 0,
  xi1 = 1/3,
  xi2 = 1/3,
  nullprop = FALSE
)
```

Arguments

`g`	One of two inputs • A matrix of genotype counts. The rows index the loci and the columns index the genotypes. • An array of genotype log-likelihoods. The rows index the loci, the columns index the individuals, and the slices index the genotypes. Log-likelihoods are base e (natural log).
`p1`	One of three inputs • A vector of parent 1's genotypes. • A matrix of parent 1's genotype log-likelihoods. The rows index the loci and the columns index the genotypes. Logs are in base e (natural log). • `NULL` (only supported when using genotype likelihoods for the offspring)
`p2`	One of three inputs • A vector of parent 1's genotypes. • A matrix of parent 1's genotype log-likelihoods. The rows index the loci and the columns index the genotypes. Logs are in base e (natural log). • `NULL` (only supported when using genotype likelihoods for the offspring)
`drbound`	The upper bound on the double reduction rate.
`pp`	Is (partial) preferential pairing possible (`TRUE`) or not (`FALSE`)?
`dr`	Is double reduction possible (`TRUE`) or not (`FALSE`)?
`alpha`	If `dr` = `FALSE`, this is the known rate of double reduction.
`xi1`	If `pp` = `FALSE`, this is the known preferential pairing parameter of parent 1.
`xi2`	If `pp` = `FALSE`, this is the known preferential pairing parameter of parent 2.
`nullprop`	Should we return the null proportions (`TRUE`) or not (`FALSE`)?

Value

A data frame with the following elements:

`statistic` The likelihood ratio test statistic

`p_value` The p-value of the likelihood ratio test.

`df` The degrees of freedom of the test.

`alpha` The MLE of the double reduction rate. Do not use for real work.

`xi1` The MLE of the first parent's partial preferential pairing parameter. Do not use for real work.

`xi2` The MLE of the second parent's partial preferential pairing parameter. Do not use for real work.

`p1` (Estimate of) the first parent's genotype.

`p2` (Estimate of) the second parent's genotype.

`snp` The name of the SNP.

Parallel Computation

The `multi_lrt()` function supports parallel computing. It does so through the future package.

You first specify the evaluation plan with `plan()` from the `future` package. On a local machine, this is typically just `future::plan(future::multisession, workers = nc)` where `nc` is the number of workers you want. You can find the maximum number of possible workers with `availableCores()`. You then run `multi_lrt()`, then shut down the workers with `future::plan(future::sequential)`.

Author(s)

David Gerard

See Also

- `lrt_men_g4()` Single locus LRT for segregation distortion when genotypes are known.
- `lrt_men_gl4()` Single locus LRT for segregation distortion when using genotype likelihoods.

Examples

```
## Assuming genotypes are known (typically a bad idea)
glist <- multidog_to_g(mout = ufit, type = "all_g", p1 = "indigocrisp", p2 = "sweetcrisp")
p1_1 <- glist$p1
p2_1 <- glist$p2
g_1 <- glist$g
multi_lrt(g = g_1, p1 = p1_1, p2 = p2_1)

## Using genotype likelihoods (typically a good idea)
glist <- multidog_to_g(mout = ufit, type = "all_gl", p1 = "indigocrisp", p2 = "sweetcrisp")
p1_2 <- glist$p1
p2_2 <- glist$p2
g_2 <- glist$g
multi_lrt(g = g_2, p1 = p1_2, p2 = p2_2)

## Offspring genotype likelihoods and parent genotypes known
multi_lrt(g = g_2, p1 = p1_1, p2 = p2_1)

## Offspring genotype likelihoods and no information on parent genotypes
multi_lrt(g = g_2, p1 = NULL, p2 = NULL)

## Parallel computing is supported through the future package
future::plan(future::multisession, workers = 2)
multi_lrt(g = g_2, p1 = p1_2, p2 = p2_2)
future::plan(future::sequential)
```

nextElem.arrayiter *Next element in an array*

Description

This is applied to an `arrayiter` object to obtain the next sub-array along one of the dimensions.

Usage

```
## S3 method for class 'arrayiter'
nextElem(obj, ...)
```

Arguments

`obj`	An arrayiter object
`...`	not used

Value

The next sub-array.

Author(s)

David Gerard

See Also

`iter.array()`

Examples

```
glist <- multidog_to_g(mout = ufit, type = "all_gl", p1 = "indigocrisp", p2 = "sweetcrisp")
g <- iterators::iter(glist$g, by = 3)
head(iterators::nextElem(g))
head(iterators::nextElem(g))
head(iterators::nextElem(g))
```

`offspring_geno` *Simulates genotypes given genotype frequencies.*

Description

Takes as input the offspring genotype frequencies and a sample size and returns simulated genotypes.

Usage

```
offspring_geno(gf, n)
```

Arguments

`gf`	Vector of offspring genotype frequencies
`n`	Sample size

Value

Simulated genotypes

Author(s)

Mira Thakkar

Examples

```
set.seed(1)
gf <- offspring_gf_2(alpha = 1/6, xi1 = 1/3, xi2 = 1/3, p1 = 2, p2 = 3)
offspring_geno(gf = gf, n = 10)
```

`offspring_gf_2` *Calculates offspring genotype frequencies under the two-parameter model.*

Description

Calculates offspring genotype frequencies under the two-parameter model.

Usage

```
offspring_gf_2(alpha, xi1, xi2 = xi1, p1, p2)
```

Arguments

`alpha`	The double reduction rate
`xi1`	The preferential pairing parameter of the first parent.
`xi2`	The preferential pairing parameter of the second parent.
`p1`	The first parent's genotype
`p2`	The second parent's genotype

Value

Offspring genotype frequencies

Author(s)

Mira Thakkar

Examples

```
alpha <- 1/6
xi1 <- 1/3
xi2 <- 1/3
p1 <- 2
p2 <- 3
offspring_gf_2(alpha = alpha, xi1 = xi1, xi2 = xi2, p1 = p1, p2 = p2)
```

`offspring_gf_3` *Calculates offspring genotype frequencies under the three-parameter model.*

Description

Calculates offspring genotype frequencies under the three-parameter model.

Usage

```
offspring_gf_3(tau, beta, gamma1, gamma2 = gamma1, p1, p2)
```

Arguments

`tau`	Probability of quadrivalent formation
`beta`	Probability of double reduction given quadrivalent formation
`gamma1`	Probability of AA_aa pairing in parent 1
`gamma2`	Probability of AA_aa pairing in parent 2
`p1`	The first parent's genotype
`p2`	The second parent's genotype

Value

Offspring genotype frequencies

Author(s)

David Gerard

Examples

```
offspring_gf_3(
  tau = 1/2,
  beta = 1/6,
  gamma1 = 1/3,
  gamma2 = 1/3,
  p1 = 1,
  p2 = 2)
```

`otest_g` *Jointly tests for segregation distortion and number of incompatible genotypes*

Description

This is experimental. I haven't tested it out in lots of scenarios yet.

Usage

```
otest_g(
  x,
  g1,
  g2,
  pbad = 0.03,
  drbound = 1/6,
  pp = TRUE,
  dr = TRUE,
  alpha = 0,
  xi1 = 1/3,
  xi2 = 1/3
)
```

Arguments

x	A vector of genotype counts. `x[i]` is the number of offspring with genotype `i-1`.
g1	The genotype of parent 1.
g2	The genotype of parent 2.

pbad	The upper bound on the number of bad genotypes
drbound	The maximum rate of double reduction. A default of 1/6 is provided, which is the rate under the complete equational segregation model of meiosis.
pp	A logical. Should we account for preferential pairing (TRUE) or not (FALSE)?
dr	A logical. Should we account for double reduction (TRUE) or not (FALSE)?
alpha	If dr = FALSE, this is the known rate of double reduction.
xi1	If pp = FALSE, this is the known preferential pairing parameter of parent 1.
xi2	If pp = FALSE, this is the known preferential pairing parameter of parent 2.

Details

Here, we test if the compatible genotypes are consistent with F1 populations and separately test that the number of incompatible genotypes isn't too large (less than 3 percent by default). This is the strategy the polymapR software uses. But we use a Bonferroni correction to combine these tests (minimum of two times the p-values), while they just multiply the p-values together. So our approach accounts for double reduction and preferential pairing, while also controlling the family-wise error rate.

Value

A list with the following elements

statistic The log-likelihood ratio test statistic.

df The degrees of freedom.

p_value The Bonferroni corrected p-value.

p_lrt The p-value of the LRT.

p_binom The p-value of the one-sided binomial test.

alpha The estimated double reduction rate.

xi1 The estimated preferential pairing parameter of parent 1.

xi2 The estimated preferential pairing parameter of parent 2.

Author(s)

David Gerard

Examples

```
# Run a test where genotypes 0, 1, and 2 are possible
x <- c(10, 10, 4, 0, 5)
otest_g(x = x, g1 = 1, g2 = 0)

# polymapR's multiplication and the Bonferroni differ
df <- expand.grid(p1 = seq(0, 1, length.out = 20), p2 = seq(0, 1, length.out = 20))
df$polymapr <- NA
df$bonferroni <- NA
for (i in seq_len(nrow(df))) {
  df$polymapr[[i]] <- df$p1[[i]] * df$p2[[i]]
  df$bonferroni[[i]] <- 2 * min(c(df$p1[[i]], df$p2[[i]], 0.5))
}
graphics::plot(df$polymapr, df$bonferroni)
```

`polymapr_test` *Run segregation distortion tests as implemented in the polymapR package.*

Description

The polymapR package tests for segregation distortion by iterating through all possible forms of disomic or polysomic inheritance from either parent, tests for concordance of the offspring genotypes using a chi-squared test, and returns the largest p-value. It sometimes chooses a different p-value based on other heuristics. They also sometimes return NA. When `type = "segtest"`, we only look at patterns of the given parent genotypes, choosing the largest p-value. When `type = "polymapR"`, we return what they use via their heuristics.

Usage

```
polymapr_test(x, g1 = NULL, g2 = NULL, type = c("segtest", "polymapR"))
```

Arguments

`x`	Either a vector of genotype counts, or a matrix of genotype posteriors where the rows index the individuals and the columns index the genotypes.
`g1`	Parent 1's genotype.
`g2`	Parent 2's genotype.
`type`	Either my implementation which approximates that of polymapR (`"segtest"`) or the implementation through polymapR (`"polymapR"`). Note that polymapR needs to be installed for `type = "polymapR"`.

Value

A list with the following elements:

p_value The p-value of the test.

bestfit The best fit model, using the same notation as in `checkF1()`.

frq_invalid The frequency of invalid genotypes.

Author(s)

David Gerard

See Also

`checkF1()`.

Examples

```
g1 <- 0
g2 <- 1
x <- c(4, 16, 0, 0, 0)
polymapr_test(x = x, g1 = g1, g2 = g2, type = "segtest")
```

po_gl   *Generate genotype likelihoods from offspring genotypes.*

Description

Takes as input (i) the parent genotypes, (ii) the offspring genotype frequency, (iii) sequencing error rate, (iv) read depth, (v) bias, and (vi) overdispersion. It returns genotype likelihoods.

Usage

```
po_gl(
  genovec,
  ploidy,
  p1_geno = NULL,
  p2_geno = NULL,
  rd = 10,
  seq = 0.01,
  bias = 1,
  od = 0.01
)
```

Arguments

`genovec`	Offspring genotypes. `genovec[i]` is the dosage for individual i.
`ploidy`	Ploidy
`p1_geno`	Parent 1 genotype
`p2_geno`	Parent 2 genotype
`rd`	Read depth. Lower is more uncertain.
`seq`	Sequencing error rate. Higher means more uncertain.
`bias`	Bias. 1 means no bias.
`od`	Overdispersion. Typical value is like 0.01. Higher means more uncertain.

Value

Genotype likelihoods

Author(s)

Mira Thakkar

Examples

```
set.seed(1)
po_gl(genovec = c(1, 2, 1, 1, 3), p1_geno = 2, p2_geno = 2, ploidy = 4)
```

`pvec_tet_2` *Tetraploid gamete frequencies of gametes when one parent's genotype is known*

Description

This is under the two parameter model.

Usage

```
pvec_tet_2(alpha, xi, ell)
```

Arguments

`alpha`	The double reduction rate
`xi`	The preferential pairing parameter
`ell`	The parental genotype

Value

The gamete genotype frequencies

Author(s)

Mira Thakkar and David Gerard

Examples

```
alpha <- 1/6
xi <- 1/3
pvec_tet_2(alpha = alpha, xi = xi, ell = 0)
pvec_tet_2(alpha = alpha, xi = xi, ell = 1)
pvec_tet_2(alpha = alpha, xi = xi, ell = 2)
pvec_tet_2(alpha = alpha, xi = xi, ell = 3)
pvec_tet_2(alpha = alpha, xi = xi, ell = 4)
```

pvec_tet_3 *Tetraploid gamete frequencies of gametes when one parent's genotype is known*

Description

This is under the three parameter model.

Usage

```
pvec_tet_3(tau, beta, gamma, ell)
```

Arguments

tau	Probability of quadrivalent formation
beta	Probability of double reduction given quadrivalent formation
gamma	Probability of AA/aa pairing given bivalent formation
ell	The parent genotype

Value

The gamete genotype frequencies

Author(s)

David Gerard

Examples

```
pvec_tet_3(tau = 0.5, beta = 0.1, gamma = 0.5, ell = 2)
```

simf1g *Simulate genotype counts from F1 individuals*

Description

Simulate genotype counts from F1 individuals

Usage

```
simf1g(n, g1, g2, alpha = 0, xi1 = 1/3, xi2 = 1/3)
```

Arguments

`n`	Sample size.
`g1`	The first parent's genotype.
`g2`	The second parent's genotype.
`alpha`	The double reduction rate.
`xi1`	The first parent's preferential pairing parameter.
`xi2`	The second parent's preferential pairing parameter.

Value

A vector of counts, where element `i` is the number of simulated individuals with genotype `i-1`.

Author(s)

David Gerard

Examples

```
set.seed(1)
simf1g(n = 10, g1 = 1, g2 = 2)
```

`simf1gl` *Simulate genotype likelihoods of F1 individuals.*

Description

Simulate genotype likelihoods of F1 individuals.

Usage

```
simf1gl(n, g1, g2, rd = 10, alpha = 0, xi1 = 1/3, xi2 = 1/3)
```

Arguments

`n`	Sample size.
`g1`	The first parent's genotype.
`g2`	The second parent's genotype.
`rd`	The read depth.
`alpha`	The double reduction rate.
`xi1`	The first parent's preferential pairing parameter.
`xi2`	The second parent's preferential pairing parameter.

Value

The matrix of offspring genotype log-likelihoods.

Author(s)

David Gerard

Examples

```
set.seed(1)
simf1gl(n = 10, g1 = 1, g2 = 2)
```

---

simgl *Simulate genotype likelihoods from genotype counts*

Description

Provide a vector of genotype counts and this will return a matrix of genotype log-likelihoods.

Usage

```
simgl(nvec, rd = 10, seq = 0.01, bias = 1, od = 0.01)
```

Arguments

`nvec`	A vector of counts. `nvec[k]` is the number of folks with a genotype of k-1.
`rd`	Read depth. Lower is more uncertain.
`seq`	Sequencing error rate. Higher means more uncertain.
`bias`	Bias. 1 means no bias.
`od`	Overdispersion. Typical value is like 0.01. Higher means more uncertain.

Value

A matrix of genotype log-likelihoods. The rows index the individuals and the columns index the genotypes. This is natural log (base e).

Author(s)

David Gerard

Examples

```
set.seed(1)
simgl(nvec = c(1, 2, 1, 1, 3))
```

`three_to_two` *Convert from three parameters to two parameters*

Description

Convert from three parameters to two parameters

Usage

```
three_to_two(tau, beta, gamma)
```

Arguments

`tau`	Probability of quadrivalent formation
`beta`	Probability of double reduction given quadrivalent formation
`gamma`	Probability of AA/aa pairing given bivalent formation

Value

A vector of length two. The first is the double reduction rate (`alpha`), and the second is the preferential pairing parameter (`xi`).

Author(s)

David Gerard

Examples

```
three_to_two(tau = 0.1, beta = 1/6, gamma = 1/4)
```

`ufit` *Genotype data from Cappai et al. (2020)*

Description

A subset of data from Cappai et al. (2020), fit using `multidog()`. This just contains a random set of 10 loci.

Usage

```
ufit

ufit2

ufit3
```

Format

An object of type `multidog` output from `multidog()`.

`ufit` Uses the `model = "norm"` option.

`ufit2` Uses the `model = "f1pp"` option.

`ufit3` Uses the `model = "f1"` option.

An object of class `multidog` of length 2.

An object of class `multidog` of length 2.

Source

doi:10.5281/zenodo.13715703

References

- Cappai, F., Amadeu, R. R., Benevenuto, J., Cullen, R., Garcia, A., Grossman, A., Ferrão, L., & Munoz, P. (2020). High-resolution linkage map and QTL analyses of fruit firmness in autotetraploid blueberry. *Frontiers in plant science*, 11, 562171. doi:10.3389/fpls.2020.562171.

Index