Package ‘qtlpoly’

March 25, 2024

Type Package

Title Random-Effect Multiple QTL Mapping in Autopolyploids

Version 0.2.4

Maintainer Gabriel de Siqueira Gesteira <gdesiqu@ncsu.edu>

Description Performs random-effect multiple interval mapping (REMIM) in full-sib families of autopolyploid species based on restricted maximum likelihood (REML) estimation and score statistics, as described in Pereira et al. (2020) <doi:10.1534/genetics.120.303080>.

License GPL-3

URL https://gabrielgesteira.github.io/QTLpoly/

BugReports https://github.com/gabrielgesteira/QTLpoly/issues

Encoding UTF-8

LazyData TRUE

LazyDataCompression xz

Depends R (>= 4.0)

Imports ggplot2 (>= 3.1), abind (>= 1.4), MASS (>= 7.3), gtools (>= 3.9.2), CompQuadForm, Matrix, RLRsim, mvtnorm, nlme, quadprog, parallel, doParallel, foreach, stats, methods, mappoly, Rcpp (>= 0.12.19)

LinkingTo Rcpp, RcppArmadillo, RcppProgress

Suggests rmarkdown, devtools, knitr

RoxygenNote 7.2.3

NeedsCompilation yes

Author Guilherme da Silva Pereira [aut] (<https://orcid.org/0000-0002-7106-8630>),
Marcelo Mollinari [ctb] (<https://orcid.org/0000-0002-7001-8498>),
Gabriel de Siqueira Gesteira [ctb, cre] (<https://orcid.org/0000-0002-4106-7346>),
Zhao-Bang Zeng [ctb] (<https://orcid.org/0000-0002-3115-1149>),
Long Qu [ctb] (R code for variance component tests using score statistics in R/varComp.R),

Giovanny Covarrubias-Pazaran [ctb] (C code for fitting mixed models with REML estimation in src/MNR.cpp)

Repository CRAN

Date/Publication 2024-03-25 20:50:06 UTC

R topics documented:

B2721 *Autotetraploid potato dataset*

Description

A dataset of the B2721 population which derived from a cross between two tetraploid potato varieties: Atlantic × B1829-5.

Usage

```
B2721
```

Format

An object of class `mappoly.data` from the package **mappoly**.

Author(s)

Marcelo Mollinari, `<mmollin@ncsu.edu>`

References

Mollinari M, Garcia AAF (2019) Linkage analysis and haplotype phasing in experimental autopolyploid populations with high ploidy level using hidden Markov models, *G3: Genes|Genomes|Genetics* 9 (10): 3297-3314. doi:10.1534/g3.119.400378

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Pereira GS, Mollinari M, Schumann MJ, Clough ME, Zeng ZB, Yencho C (2021) The recombination landscape and multiple QTL mapping in a *Solanum tuberosum* cv. 'Atlantic'-derived F_1 population. Heredity. doi:10.1038/s4143702100416x.

Examples

```
library(mappoly)
print(B2721)
```

`breeding_values` *Prediction of QTL-based breeding values from REMIM model*

Description

Computes breeding values for each genotyped individual based on multiple QTL models

Usage

```
breeding_values(data, fitted)

## S3 method for class 'qtlpoly.bvalues'
plot(x, pheno.col = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`fitted`	an object of class `qtlpoly.fitted`.
`x`	an object of class `qtlpoly.bvalues` to be plotted.
`pheno.col`	a numeric vector with the phenotype column numbers to be plotted; if `NULL`, all phenotypes from `'data'` will be included.
`...`	currently ignored

Value

An object of class `qtlpoly.bvalues` which is a list of `results` for each trait containing the following components:

`pheno.col`	a phenotype column number.
`y.hat`	a column matrix of breeding value for each individual.

A **ggplot2** histogram with the distribution of breeding values.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`read_data`, `fit_model`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob) #5,7
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Fit model
 fitted.mod = fit_model(data = data, model = remim.mod, probs = "joint", polygenes = "none")

  # Predict genotypic values
  y.hat = breeding_values(data = data, fitted = fitted.mod)
  plot(y.hat)
```

`feim` *Fixed-effect interval mapping (FEIM)*

Description

Performs interval mapping using the single-QTL, fixed-effect model proposed by Hackett et al. (2001).

Usage

```
feim(
  data = data,
  pheno.col = NULL,
  w.size = 15,
  sig.lod = 7,
  d.sint = 1.5,
  plot = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.feim'
print(x, pheno.col = NULL, sint = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`pheno.col`	a numeric vector with the phenotype columns to be analyzed; if `NULL` (default), all phenotypes from `'data'` will be included.
`w.size`	a number representing the window size (in centiMorgans) to be avoided on either side of QTL already in the model when looking for a new QTL, e.g. 15 (default).
`sig.lod`	the vector of desired significance LOD thresholds (usually permutation-based) for declaring a QTL for each trait, e.g. 5 (default); if a single value is provided, the same LOD threshold will be applied to all traits.
`d.sint`	a d value to subtract from logarithm of the odds ($LOD - d$) for support interval calculation, e.g. $d = 1.5$ (default) represents approximate 95% support interval.
`plot`	a suffix for the file's name containing plots of every algorithm step, e.g. "remim" (default); if `NULL`, no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.feim` to be printed.
`sint`	whether `"upper"` or `"lower"` support intervals should be printed; if `NULL` (default), QTL peak information will be printed.
`...`	currently ignored

Value

An object of class `qtlpoly.feim` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`LRT`	a vector containing LRT values.
`LOD`	a vector containing LOD scores.
`AdjR2`	a vector containing adjusted R^2.
`qtls`	a data frame with information from the mapped QTL.
`lower`	a data frame with information from the lower support interval of mapped QTL.
`upper`	a data frame with information from the upper support interval of mapped QTL.

Author(s)

Guilherme da Silva Pereira, `<gdasilv@ncsu.edu>`

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Hackett CA, Bradshaw JE, McNicol JW (2001) Interval mapping of quantitative trait loci in autotetraploid species, *Genetics* 159: 1819-1832.

See Also

`permutations`

Examples

```
# Estimate conditional probabilities using mappoly package
library(mappoly)
library(qtlpoly)
genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 5)

# Perform remim
feim.mod = feim(data = data, sig.lod = 7)
```

fit_model *Fits multiple QTL models*

Description

Fits alternative multiple QTL models by performing variance component estimation using REML.

Usage

```
fit_model(
  data,
  model,
  probs = "joint",
  polygenes = "none",
  keep = TRUE,
  verbose = TRUE,
  pheno.col = NULL
)

## S3 method for class 'qtlpoly.fitted'
summary(object, pheno.col = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`model`	an object of class `qtlpoly.profile` or `qtlpoly.remim`.
`probs`	a character string indicating if either `"joint"` (genotypes) or `"marginal"` (parental gametes) conditional probabilities should be used.
`polygenes`	a character string indicating if either `"none"`, `"most"` or `"all"` QTL should be used as polygenes.
`keep`	if `TRUE` (default), stores all matrices and estimates from fitted model; if `FALSE`, nothing is stored.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`pheno.col`	a numeric vector with the phenotype column numbers to be summarized; if `NULL` (default), all phenotypes from `'data'` will be included.
`object`	an object of class `qtlpoly.fitted` to be summarized.
`...`	currently ignored

Value

An object of class `qtlpoly.fitted` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`fitted`	a **sommer** object of class `mmer`.
`qtls`	a data frame with information from the mapped QTL.

Author(s)

Guilherme da Silva Pereira, <gdasilv@ncsu.edu>

References

Covarrubias-Pazaran G (2016) Genome-assisted prediction of quantitative traits using the R package sommer. *PLoS ONE* 11 (6): 1–15. doi:10.1371/journal.pone.0156744.

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`read_data`, `remim`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Fit model
  fitted.mod = fit_model(data=data, model=remim.mod, probs="joint", polygenes="none")
```

`fit_model2` *Fits multiple QTL models*

Description

Fits alternative multiple QTL models by performing variance component estimation using REML.

Usage

```
fit_model2(
  data,
  model,
  probs = "joint",
  polygenes = "none",
  keep = TRUE,
  verbose = TRUE,
  pheno.col = NULL
)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`model`	an object of class `qtlpoly.profile` or `qtlpoly.remim`.
`probs`	a character string indicating if either `"joint"` (genotypes) or `"marginal"` (parental gametes) conditional probabilities should be used.
`polygenes`	a character string indicating if either `"none"`, `"most"` or `"all"` QTL should be used as polygenes.
`keep`	if `TRUE` (default), stores all matrices and estimates from fitted model; if `FALSE`, nothing is stored.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`pheno.col`	a numeric vector with the phenotype column numbers to be summarized; if `NULL` (default), all phenotypes from `'data'` will be included.

Value

An object of class `qtlpoly.fitted` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`fitted`	a **sommer** object of class `mmer`.
`qtls`	a data frame with information from the mapped QTL.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Covarrubias-Pazaran G (2016) Genome-assisted prediction of quantitative traits using the R package sommer. *PLoS ONE* 11 (6): 1–15. doi:10.1371/journal.pone.0156744.

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`read_data`, `remim`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Fit model
  fitted.mod = fit_model(data=data, model=remim.mod, probs="joint", polygenes="none")
```

`hexafake` *Simulated autohexaploid dataset.*

Description

A dataset of a hypothetical autohexaploid full-sib population containing three homology groups

Usage

`hexafake`

Format

An object of class `mappoly.data` which contains a list with the following components:

plody ploidy level = 6

n.ind number individuals = 300

n.mrk total number of markers = 1500

ind.names the names of the individuals

mrk.names the names of the markers

dosage.p1 a vector containing the dosage in parent P for all `n.mrk` markers

dosage.p2 a vector containing the dosage in parent Q for all `n.mrk` markers

chrom a vector indicating the chromosome each marker belongs. Zero indicates that the marker was not assigned to any chromosome

genome.pos Physical position of the markers into the sequence

geno.dose a matrix containing the dosage for each markers (rows) for each individual (columns). Missing data are represented by `ploidy_level + 1 = 7`

n.phen There are no phenotypes in this simulation

phen There are no phenotypes in this simulation

chisq.pval vector containing p-values for all markers associated to the chi-square test for the expected segregation patterns under Mendelian segregation

Author(s)

Marcelo Mollinari, `<mmollin@ncsu.edu>`

References

Mollinari M, Garcia AAF (2019) Linkage analysis and haplotype phasing in experimental autopolyploid populations with high ploidy level using hidden Markov models, *G3: Genes|Genomes|Genetics* 9 (10): 3297-3314. doi:10.1534/g3.119.400378

Examples

```
library(mappoly)
plot(hexafake)
```

`maps4x` *Autotetraploid potato map*

Description

A real autotetraploid potato map containing 12 homology groups from a tetraploid potato full-sib family (Atlantic x B1829-5).

Usage

```
maps4x
```

Format

An object of class `"mappoly.map"` from the package **mappoly**, which is a list of 12 linkage groups (LGs)

Author(s)

Marcelo Mollinari, `<mmollin@ncsu.edu>`

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2019) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Mollinari M, Garcia AAF (2019) Linkage analysis and haplotype phasing in experimental autopolyploid populations with high ploidy level using hidden Markov models, *G3: Genes|Genomes|Genetics* 9 (10): 3297-3314. doi:10.1534/g3.119.400378

Pereira GS, Mollinari M, Schumann MJ, Clough ME, Zeng ZB, Yencho C (2021) The recombination landscape and multiple QTL mapping in a *Solanum tuberosum* cv. 'Atlantic'-derived F_1 population. Heredity. doi:10.1038/s4143702100416x.

See Also

hexafake, pheno6x

Examples

```
library(mappoly)
plot_map_list(maps4x)
```

---

maps6x *Simulated autohexaploid map*

Description

A simulated map containing three homology groups of a hypotetical cross between two autohexaploid individuals.

Usage

```
maps6x
```

Format

An object of class `"mappoly.map"` from the package **mappoly**, which is a list of three linkage groups (LGs):

LG 1	538 markers distributed along 112.2 cM
LG 2	329 markers distributed along 54.6 cM
LG 3	443 markers distributed along 98.2 cM

Author(s)

Marcelo Mollinari, `<mmollin@ncsu.edu>`

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2019) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Mollinari M, Garcia AAF (2019) Linkage analysis and haplotype phasing in experimental autopolyploid populations with high ploidy level using hidden Markov models, *G3: Genes|Genomes|Genetics* 9 (10): 3297-3314. doi:10.1534/g3.119.400378

See Also

hexafake, pheno6x

Examples

```
library(mappoly)
plot_map_list(maps6x)
```

`modify_qtl` *Modify QTL model*

Description

Adds or removes QTL manually from a given model.

Usage

```
modify_qtl(
  model,
  pheno.col = NULL,
  add.qtl = NULL,
  drop.qtl = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.modify'
print(x, pheno.col = NULL, ...)
```

Arguments

`model`	an object of class `qtlpoly.model` containing the QTL to be modified.
`pheno.col`	a phenotype column number whose model will be modified or printed.
`add.qtl`	a marker position number to be added.
`drop.qtl`	a marker position number to be removed.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.modify` to be printed.
`...`	currently ignored

Value

An object of class `qtlpoly.modify` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing *p*-values from score statistics.
`qtls`	a data frame with information from the mapped QTL.

Author(s)

Guilherme da Silva Pereira, <gdasilv@ncsu.edu>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

read_data, remim

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Modify model
  modified.mod = modify_qtl(model = remim.mod, pheno.col = 1, drop.qtl = 18)
```

---

null_model *Null model*

Description

Creates a null model (with no QTL) for each trait.

Usage

```
null_model(
  data,
  offset.data = NULL,
  pheno.col = NULL,
  n.clusters = NULL,
  plot = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.null'
print(x, pheno.col = NULL, ...)

## S3 method for class 'qtlpoly.null'
print(x, pheno.col = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`offset.data`	a data frame with the same dimensions of `data$pheno` containing offset variables; if `NULL` (default), no offset variables are considered.
`pheno.col`	a numeric vector with the phenotype columns to be analyzed; if `NULL`, all phenotypes from `'data'` will be included.
`n.clusters`	number of parallel processes to spawn.
`plot`	a suffix for the file's name containing simple plots of every QTL search round, e.g. "null" (default); if `NULL`, no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.null` to be printed.
`...`	currently ignored

Value

An object of class `qtlpoly.null` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing *p*-values from score statistics.
`qtls`	a data frame with information from the mapped QTL (`NULL` at this point).

Author(s)

Guilherme da Silva Pereira, <gdasilv@ncsu.edu>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

See Also

read_data

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Build null models
  null.mod = null_model(data = data, pheno.col = 1, n.clusters = 1)
```

---

null_model2 *Null model*

Description

Creates a null model (with no QTL) for each trait.

Usage

```
null_model2(
  data,
  offset.data = NULL,
  pheno.col = NULL,
  n.clusters = NULL,
  plot = NULL,
  verbose = TRUE
)
```

Arguments

data	an object of class qtlpoly.data.
offset.data	a data frame with the same dimensions of data$pheno containing offset variables; if NULL (default), no offset variables are considered.
pheno.col	a numeric vector with the phenotype columns to be analyzed; if NULL, all phenotypes from 'data' will be included.
n.clusters	number of parallel processes to spawn.
plot	a suffix for the file's name containing simple plots of every QTL search round, e.g. "null" (default); if NULL, no file is produced.
verbose	if TRUE (default), current progress is shown; if FALSE, no output is produced.

Value

An object of class `qtlpoly.null` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing *p*-values from score statistics.
`qtls`	a data frame with information from the mapped QTL (`NULL` at this point).

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>, Gabriel de Siqueira Gesteira, <`gdesiqu@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

See Also

`read_data`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Build null models
  null.mod = null_model(data = data, pheno.col = 1, n.clusters = 1)
```

optimize_qtl *Model optimization*

Description

Tests each QTL at a time and updates its position (if it changes) or drops the QTL (if non-significant).

Usage

```
optimize_qtl(
  data,
  offset.data = NULL,
  model,
  sig.bwd = 0.05,
  score.null = NULL,
  polygenes = FALSE,
  n.clusters = NULL,
  plot = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.optimize'
print(x, pheno.col = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`offset.data`	a data frame with the same dimensions of `data$pheno` containing offset variables; if `NULL` (default), no offset variables are considered.
`model`	an object of class `qtlpoly.model` containing the QTL to be optimized.
`sig.bwd`	the desired score-based *p*-value threshold for backward elimination, e.g. 0.0001 (default).
`score.null`	an object of class `qtlpoly.null` with results of score statistics from resampling.
`polygenes`	if `TRUE` all QTL but the one being tested are treated as a single polygenic effect, if `FALSE` (default) all QTL effect variances have to estimated.
`n.clusters`	number of parallel processes to spawn.
`plot`	a suffix for the file's name containing plots of every QTL optimization round, e.g. "optimize" (default); if `NULL`, no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.optimize` to be printed.
`pheno.col`	a numeric vector with the phenotype columns to be printed; if `NULL`, all phenotypes from `'data'` will be included.
`...`	currently ignored

Value

An object of class `qtlpoly.optimize` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing *p*-values from score statistics.
`qtls`	a data frame with information from the mapped QTL.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

Zou F, Fine JP, Hu J, Lin DY (2004) An efficient resampling method for assessing genome-wide statistical significance in mapping quantitative trait loci. *Genetics* 168 (4): 2307-16. doi:10.1534/genetics.104.031427

See Also

`read_data`, `null_model`, `search_qtl`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Build null model
  null.mod = null_model(data = data, pheno.col = 1,n.clusters = 1)

  # Perform forward search
  search.mod = search_qtl(data = data, model = null.mod,
w.size = 15, sig.fwd = 0.01, n.clusters = 1)

  # Optimize model
 optimize.mod = optimize_qtl(data = data, model = search.mod, sig.bwd = 0.0001, n.clusters = 1)
```

permutations	*Fixed-effect interval mapping (FEIM) model permutations*

Description

Stores maximum LOD scores for a number of permutations of given phenotypes.

Usage

```
permutations(
  data,
  offset.data = NULL,
  pheno.col = NULL,
  n.sim = 1000,
  probs = c(0.9, 0.95),
  n.clusters = NULL,
  seed = 123,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.perm'
print(x, pheno.col = NULL, probs = c(0.9, 0.95), ...)

## S3 method for class 'qtlpoly.perm'
plot(x, pheno.col = NULL, probs = c(0.9, 0.95), ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`offset.data`	a subset of the data object to be used in permutation calculations.
`pheno.col`	a numeric vector with the phenotype columns to be analyzed; if `NULL` (default), all phenotypes from `'data'` will be included.
`n.sim`	a number of simulations, e.g. 1000 (default).
`probs`	a vector of probability values in [0, 1] representing the quantiles, e.g. c(0.90, 0.95) for the 90% and 95% quantiles.
`n.clusters`	a number of parallel processes to spawn.
`seed`	an integer for the `set.seed()` function; if `NULL`, no reproducible seeds are set.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.perm` to be printed or plotted.
`...`	currently ignored

Value

An object of class `qtlpoly.perm` which contains a list of `results` for each trait with the maximum LOD score per permutation.

LOD score thresholds for given quantiles for each trait.

A **ggplot2** histogram with the distribution of ordered maximum LOD scores and thresholds for given quantiles for each trait.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Churchill GA, Doerge RW (1994) Empirical threshold values for quantitative trait mapping, *Genetics* 138: 963-971.

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`feim`

Examples

```
# Estimate conditional probabilities using mappoly package
library(mappoly)
library(qtlpoly)
genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

# Perform permutations
perm = permutations(data = data, pheno.col = 1, n.sim = 10, n.clusters = 1)
```

`pheno4x` *Autotetraploid potato phenotypes*

Description

A subset of phenotypes from a tetraploid potato full-sib family (Atlantic x B1829-5).

Usage

`pheno4x`

Format

A data frame of phenotypes with 156 named individuals in rows and three named phenotypes in columns, which are:

FM07 Foliage maturity evaluated in 2007.

FM08 Foliage maturity evaluated in 2008.

FM14 Foliage maturity evaluated in 2014.

Author(s)

Guilherme da Silva Pereira, `<gdasilv@ncsu.edu>`

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Pereira GS, Mollinari M, Schumann MJ, Clough ME, Zeng ZB, Yencho C (2021) The recombination landscape and multiple QTL mapping in a *Solanum tuberosum* cv. 'Atlantic'-derived F_1 population. Heredity. doi:10.1038/s41437021004l6x.

Examples

```
head(pheno4x)
```

`pheno6x` *Simulated phenotypes*

Description

A simulated data set of phenotypes for a hipotetical autohexaploid species map.

Usage

```
pheno6x
```

Format

A data frame of phenotypes with 300 named individuals in rows and three named phenotypes in columns, which are:

T32 3 QTLs, with heritabilities of 0.20 (LG 1 at 32.03 cM), 0.15 (LG 1 at 95.02 cM) and 0.30 (LG 2 at 40.01 cM).

T17 1 QTL, with heritability of 0.15 (LG 3 at 34.51 cM).

T45 no QTLs.

Author(s)

Guilherme da Silva Pereira, <gdasilv@ncsu.edu>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2019) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`simulate_qtl`, `pheno4x`

Examples

```
head(pheno6x)
```

`plot_profile` *Logarithm of* P*-value (LOP) profile plots*

Description

Plots profiled logarithm of score-based *P*-values (LOP) from individual or combined traits.

Usage

```
plot_profile(
  data = data,
  model = model,
  pheno.col = NULL,
  sup.int = FALSE,
  main = NULL,
  legend = "bottom",
  ylim = NULL,
  grid = FALSE
)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`model`	an object of class `qtlpoly.profile` or `qtlpoly.remim`.
`pheno.col`	a numeric vector with the phenotype column numbers to be plotted; if `NULL`, all phenotypes from `'data'` will be included.
`sup.int`	if `TRUE`, support interval are shown as shaded areas; if `FALSE` (default), no support interval is show.

`main`	a character string with the main title; if `NULL`, no title is shown.
`legend`	legend position (either "bottom", "top", "left" or "right"); if `NULL`, no legend is shown.
`ylim`	a numeric value pair supplying the limits of y-axis, e.g. c(0,10); if `NULL` (default), limits will be provided automatically.
`grid`	if `TRUE`, profiles will be organized in rows (one per trait); if `FALSE` (default), profiles will appear superimposed. Only effective when plotting profiles from more than one trait.

Value

A **ggplot2** with the LOP profiles for each trait.

Author(s)

Guilherme da Silva Pereira, <gdasilv@ncsu.edu>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

profile_qtl, remim

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Plot profile
  plot_profile(data = data, model = remim.mod, grid = FALSE)
```

`plot_qtl`	*QTL heritability and significance plot*

Description

Creates a plot where dot sizes and colors represent the QTLs heritabilities and their *p*-values, respectively.

Usage

```
plot_qtl(
  data = data,
  model = model,
  fitted = fitted,
  pheno.col = NULL,
  main = NULL,
  drop.pheno = TRUE,
  drop.lgs = TRUE
)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`model`	an object of class `qtlpoly.profile` or `qtlpoly.remim`.
`fitted`	an object of class `qtlpoly.fitted`.
`pheno.col`	the desired phenotype column numbers to be plotted. The order here specifies the order of plotting (from top to bottom.)
`main`	plot title; if `NULL` (the default), no title is shown.
`drop.pheno`	if `FALSE`, shows the names of all traits from `pheno.col`, even of those with no QTLs; if `TRUE` (the default), shows only the traits with QTL(s).
`drop.lgs`	if `FALSE`, shows all linkage groups, even those with no QTL; if `TRUE` (the default), shows only the linkage groups with QTL(s).

Value

A **ggplot2** with dots representing the QTLs.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

read_data, remim, fit_model

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Fit model
  fitted.mod = fit_model(data, remim.mod, probs="joint", polygenes="none")

  # Plot QTL
  plot_qtl(data, remim.mod, fitted.mod)
```

---

plot_sint — *QTLs with respective support interval plots*

Description

Creates a plot where colored bars represent the support intervals for QTL peaks (black dots).

Usage

```
plot_sint(data, model, pheno.col = NULL, main = NULL, drop = FALSE)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`model`	an object of class `qtlpoly.profile` or `qtlpoly.remim`.
`pheno.col`	a numeric vector with the phenotype column numbers to be plotted; if `NULL`, all phenotypes from `'data'` will be included.
`main`	a character string with the main title; if `NULL`, no title will be shown.
`drop`	if `TRUE`, phenotypes with no QTL will be dropped; if `FALSE` (default), all phenotypes will be shown.

Value

A **ggplot2** with QTL bars for each linkage group.

Author(s)

Guilherme da Silva Pereira, <gdasilv@ncsu.edu>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

read_data, remim, profile_qtl

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Plot support intervals
  plot_sint(data = data, model = remim.mod)
```

---

profile_qtl *QTL profiling*

Description

Generates the score-based genome-wide profile conditional to the selected QTL.

Usage

```
profile_qtl(
  data,
  model,
  d.sint = 1.5,
  polygenes = FALSE,
  n.clusters = NULL,
  plot = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.profile'
print(x, pheno.col = NULL, sint = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`model`	an object of class `qtlpoly.model` containing the QTL to be profiled.
`d.sint`	a d value to subtract from logarithm of *p*-value ($LOP - d$) for support interval calculation, e.g. $d = 1.5$ (default) represents approximate 95% support interval.
`polygenes`	if `TRUE` all QTL but the one being tested are treated as a single polygenic effect, if `FALSE` (default) all QTL effect variances have to estimated.
`n.clusters`	number of parallel processes to spawn.
`plot`	a suffix for the file's name containing plots of every QTL profiling round, e.g. "profile" (default); if `NULL`, no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.profile` to be printed.
`pheno.col`	a numeric vector with the phenotype column numbers to be plotted; if `NULL`, all phenotypes from `'data'` will be included.
`sint`	whether `"upper"` or `"lower"` support intervals should be printed; if `NULL` (default), only QTL peak information will be printed.
`...`	currently ignored

Value

An object of class `qtlpoly.profile` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing *p*-values from score statistics.
`qtls`	a data frame with information from the mapped QTL.
`lower`	a data frame with information from the lower support interval of mapped QTL.
`upper`	a data frame with information from the upper support interval of mapped QTL.

Author(s)

Guilherme da Silva Pereira, `<gdasilv@ncsu.edu>`

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Build null model
  null.mod = null_model(data, pheno.col = 1, n.clusters = 1)

  # Perform forward search
  search.mod = search_qtl(data = data, model = null.mod,
w.size = 15, sig.fwd = 0.01, n.clusters = 1)

  # Optimize model
 optimize.mod = optimize_qtl(data = data, model = search.mod, sig.bwd = 0.0001, n.clusters = 1)

  # Profile model
 profile.mod = profile_qtl(data = data, model = optimize.mod, d.sint = 1.5, n.clusters = 1)
```

`qtl_effects` *QTL allele effect estimation*

Description

Computes allele specific and allele combination (within-parent) heritable effects from multiple QTL models.

Usage

```
qtl_effects(ploidy = 6, fitted, pheno.col = NULL, verbose = TRUE)

## S3 method for class 'qtlpoly.effects'
plot(x, pheno.col = NULL, p1 = "P1", p2 = "P2", ...)
```

Arguments

`ploidy`	a numeric value of ploidy level of the cross (currently, only 2, 4 or 6).
`fitted`	a fitted multiple QTL model of class `qtlpoly.fitted`.
`pheno.col`	a numeric vector with the phenotype column numbers to be plotted; if `NULL`, all phenotypes from `'fitted'` will be included.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.effects` to be plotted.
`p1`	a character string with the first parent name, e.g. `"P1"` (default).
`p2`	a character string with the second parent name, e.g. `"P2"` (default).
`...`	currently ignored

Value

An object of class `qtlpoly.effects` which is a list of `results` for each containing the following components:

`pheno.col`	a phenotype column number.
`y.hat`	a vector with the predicted values.

A **ggplot2** barplot with parental allele and allele combination effects.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>, with modifications by Gabriel Gesteira, <`gdesiqu@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Kempthorne O (1955) The correlation between relatives in a simple autotetraploid population, *Genetics* 40: 168-174.

See Also

`read_data`, `remim`, `fit_model`

Examples

```
# Estimate conditional probabilities using mappoly package
library(mappoly)
library(qtlpoly)
genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)
```

```
  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)

  # Fit model
  fitted.mod = fit_model(data, model=remim.mod, probs="joint", polygenes="none")

  # Estimate effects
  est.effects = qtl_effects(ploidy = 4, fitted = fitted.mod, pheno.col = 1)

  # Plot results
  plot(est.effects)
```

read_data *Read genotypic and phenotypic data*

Description

Reads files in specific formats and creates a `qtlpoly.data` object to be used in subsequent analyses.

Usage

```
read_data(
  ploidy = 6,
  geno.prob,
  geno.dose = NULL,
  double.reduction = FALSE,
  pheno,
  weights = NULL,
  step = 1,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.data'
print(x, detailed = FALSE, ...)
```

Arguments

`ploidy`	a numeric value of ploidy level of the cross.
`geno.prob`	an object of class `mappoly.genoprob` from **mappoly**.
`geno.dose`	an object of class `mappoly.data` from **mappoly**.
`double.reduction`	if `TRUE`, double reduction genotypes are taken into account; if `FALSE`, no double reduction genotypes are considered.
`pheno`	a data frame of phenotypes (columns) with individual names (rows) identical to individual names in `geno.prob` and/or `geno.dose` object.

`weights`	a data frame of phenotype weights (columns) with individual names (rows) identical to individual names in `pheno` object.
`step`	a numeric value of step size (in centiMorgans) where tests will be performed, e.g. 1 (default); if `NULL`, tests will be performed at every marker.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.data` to be printed.
`detailed`	if `TRUE`, detailed information on linkage groups and phenotypes in shown; if `FALSE`, no details are printed.
`...`	currently ignored

Value

An object of class `qtlpoly.data` which is a list containing the following components:

`ploidy`	a scalar with ploidy level.
`nlgs`	a scalar with the number of linkage groups.
`nind`	a scalar with the number of individuals.
`nmrk`	a scalar with the number of marker positions.
`nphe`	a scalar with the number of phenotypes.
`lgs.size`	a vector with linkage group sizes.
`cum.size`	a vector with cumulative linkage group sizes.
`lgs.nmrk`	a vector with number of marker positions per linkage group.
`cum.nmrk`	a vector with cumulative number of marker positions per linkage group.
`lgs`	a list with selected marker positions per linkage group.
`lgs.all`	a list with all marker positions per linkage group.
`step`	a scalar with the step size.
`pheno`	a data frame with phenotypes.
`G`	a list of relationship matrices for each marker position.
`Z`	a list of conditional probability matrices for each marker position for genotypes.
`X`	a list of conditional probability matrices for each marker position for alleles.
`Pi`	a matrix of identical-by-descent shared alleles among genotypes.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>, with minor updates by Gabriel de Siqueira Gesteira, <`gdesiqu@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

maps6x, pheno6x

Examples

```
# Estimate conditional probabilities using mappoly package
library(mappoly)
library(qtlpoly)
genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)
```

`read_data2` *Read genotypic and phenotypic data*

Description

Reads files in specific formats and creates a `qtlpoly.data` object to be used in subsequent analyses.

Usage

```
read_data2(
  ploidy = 6,
  geno.prob,
  geno.dose = NULL,
  double.reduction = FALSE,
  pheno,
  weights = NULL,
  step = 1,
  verbose = TRUE
)
```

Arguments

`ploidy`	a numeric value of ploidy level of the cross.
`geno.prob`	an object of class `mappoly.genoprob` from **mappoly**.
`geno.dose`	an object of class `mappoly.data` from **mappoly**.
`double.reduction`	if `TRUE`, double reduction genotypes are taken into account; if `FALSE`, no double reduction genotypes are considered.
`pheno`	a data frame of phenotypes (columns) with individual names (rows) identical to individual names in `geno.prob` and/or `geno.dose` object.
`weights`	a data frame of phenotype weights (columns) with individual names (rows) identical to individual names in pheno object.
`step`	a numeric value of step size (in centiMorgans) where tests will be performed, e.g. 1 (default); if `NULL`, tests will be performed at every marker.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.

Value

An object of class `qtlpoly.data` which is a list containing the following components:

`ploidy`	a scalar with ploidy level.
`nlgs`	a scalar with the number of linkage groups.
`nind`	a scalar with the number of individuals.
`nmrk`	a scalar with the number of marker positions.
`nphe`	a scalar with the number of phenotypes.
`lgs.size`	a vector with linkage group sizes.
`cum.size`	a vector with cumulative linkage group sizes.
`lgs.nmrk`	a vector with number of marker positions per linkage group.
`cum.nmrk`	a vector with cumulative number of marker positions per linkage group.
`lgs`	a list with selected marker positions per linkage group.
`lgs.all`	a list with all marker positions per linkage group.
`step`	a scalar with the step size.
`pheno`	a data frame with phenotypes.
`G`	a list of relationship matrices for each marker position.
`Z`	a list of conditional probability matrices for each marker position for genotypes.
`X`	a list of conditional probability matrices for each marker position for alleles.
`Pi`	a matrix of identical-by-descent shared alleles among genotypes.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>, Gabriel de Siqueira Gesteira, <`gdesiqu@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`maps6x`, `pheno6x`

Examples

```
# Estimate conditional probabilities using mappoly package
library(mappoly)
library(qtlpoly)
genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)
```

remim *Random-effect multiple interval mapping (REMIM)*

Description

Automatic function that performs REMIM algorithm using score statistics.

Usage

```
remim(
  data,
  pheno.col = NULL,
  w.size = 15,
  sig.fwd = 0.01,
  sig.bwd = 1e-04,
  score.null = NULL,
  d.sint = 1.5,
  polygenes = FALSE,
  n.clusters = NULL,
  n.rounds = Inf,
  plot = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.remim'
print(x, pheno.col = NULL, sint = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`pheno.col`	a numeric vector with the phenotype columns to be analyzed or printed; if `NULL` (default), all phenotypes from `'data'` will be included.
`w.size`	the window size (in centiMorgans) to avoid on either side of QTL already in the model when looking for a new QTL, e.g. 15 (default).
`sig.fwd`	the desired score-based significance level for forward search, e.g. 0.01 (default).
`sig.bwd`	the desired score-based significance level for backward elimination, e.g. 0.001 (default).
`score.null`	an object of class `qtlpoly.null` with results of score statistics from resampling.
`d.sint`	a d value to subtract from logarithm of p-value ($LOP - d$) for support interval calculation, e.g. $d = 1.5$ (default) represents approximate 95% support interval.
`polygenes`	if `TRUE` all QTL already in the model are treated as a single polygenic effect; if `FALSE` (default) all QTL effect variances have to estimated.
`n.clusters`	number of parallel processes to spawn.
`n.rounds`	number of search rounds; if `Inf` (default) forward search will stop when no more significant positions can be found.

`plot`	a suffix for the file's name containing plots of every algorithm step, e.g. "remim"; if `NULL` (default), no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.remim` to be printed.
`sint`	whether `"upper"` or `"lower"` support intervals should be printed; if `NULL` (default), only QTL peak information will be printed.
`...`	currently ignored

Value

An object of class `qtlpoly.remim` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing p-values from score statistics.
`qtls`	a data frame with information from the mapped QTL.
`lower`	a data frame with information from the lower support interval of mapped QTL.
`upper`	a data frame with information from the upper support interval of mapped QTL.

Author(s)

Guilherme da Silva Pereira, `<gdasilv@ncsu.edu>`

References

Kao CH, Zeng ZB, Teasdale RD (1999) Multiple interval mapping for quantitative trait loci. *Genetics* 152 (3): 1203–16.

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

Zou F, Fine JP, Hu J, Lin DY (2004) An efficient resampling method for assessing genome-wide statistical significance in mapping quantitative trait loci. *Genetics* 168 (4): 2307-16. doi:10.1534/genetics.104.031427

See Also

`read_data`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)
```

---

remim2 *Random-effect multiple interval mapping (REMIM)*

Description

Automatic function that performs REMIM algorithm using score statistics.

Usage

```
remim2(
  data,
  pheno.col = NULL,
  w.size = 15,
  sig.fwd = 0.01,
  sig.bwd = 1e-04,
  score.null = NULL,
  d.sint = 1.5,
  polygenes = FALSE,
  n.clusters = NULL,
  n.rounds = Inf,
  plot = NULL,
  verbose = TRUE
)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`pheno.col`	a numeric vector with the phenotype columns to be analyzed or printed; if `NULL` (default), all phenotypes from `'data'` will be included.
`w.size`	the window size (in centiMorgans) to avoid on either side of QTL already in the model when looking for a new QTL, e.g. 15 (default).
`sig.fwd`	the desired score-based significance level for forward search, e.g. 0.01 (default).

`sig.bwd`	the desired score-based significance level for backward elimination, e.g. 0.001 (default).
`score.null`	an object of class `qtlpoly.null` with results of score statistics from resampling.
`d.sint`	a d value to subtract from logarithm of p-value ($LOP - d$) for support interval calculation, e.g. $d = 1.5$ (default) represents approximate 95% support interval.
`polygenes`	if `TRUE` all QTL already in the model are treated as a single polygenic effect; if `FALSE` (default) all QTL effect variances have to estimated.
`n.clusters`	number of parallel processes to spawn.
`n.rounds`	number of search rounds; if `Inf` (default) forward search will stop when no more significant positions can be found.
`plot`	a suffix for the file's name containing plots of every algorithm step, e.g. "remim"; if `NULL` (default), no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`sint`	whether `"upper"` or `"lower"` support intervals should be printed; if `NULL` (default), only QTL peak information will be printed.

Value

An object of class `qtlpoly.remim` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing p-values from score statistics.
`qtls`	a data frame with information from the mapped QTL.
`lower`	a data frame with information from the lower support interval of mapped QTL.
`upper`	a data frame with information from the upper support interval of mapped QTL.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>, Getúlio Caixeta Ferreira, <`getulio.caifer@gmail.com`>

References

Kao CH, Zeng ZB, Teasdale RD (1999) Multiple interval mapping for quantitative trait loci. *Genetics* 152 (3): 1203–16.

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

Zou F, Fine JP, Hu J, Lin DY (2004) An efficient resampling method for assessing genome-wide statistical significance in mapping quantitative trait loci. *Genetics* 168 (4): 2307-16. doi:10.1534/genetics.104.031427

See Also

`read_data`

Examples

```
  # Estimate conditional probabilities using mappoly package
  library(mappoly)
  library(qtlpoly)
  genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
  data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

  # Search for QTL
  remim.mod = remim2(data = data, pheno.col = 1, w.size = 15, sig.fwd = 0.0011493379,
sig.bwd = 0.0002284465, d.sint = 1.5, n.clusters = 1)
```

`search_qtl` *QTL forward search*

Description

Searches for QTL and adds them one at a time to a multiple random-effect QTL model based on score statistics.

Usage

```
search_qtl(
  data,
  offset.data = NULL,
  model,
  w.size = 15,
  sig.fwd = 0.2,
  score.null = NULL,
  polygenes = FALSE,
  n.rounds = Inf,
  n.clusters = NULL,
  plot = NULL,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.search'
print(x, pheno.col = NULL, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`offset.data`	a data frame with the same dimensions of `data$pheno` containing offset variables; if `NULL` (default), no offset variables are considered.
`model`	an object of class `qtlpoly.model` from which a forward search will start.
`w.size`	the window size (in cM) to avoid on either side of QTL already in the model when looking for a new QTL.
`sig.fwd`	the desired score-based *p*-value threshold for forward search, e.g. 0.01 (default).
`score.null`	an object of class `qtlpoly.null` with results of score statistics from resampling.
`polygenes`	if `TRUE` all QTL but the one being tested are treated as a single polygenic effect; if `FALSE` (default) all QTL effect variances have to estimated.
`n.rounds`	number of search rounds; if `Inf` (default) forward search will stop when no more significant positions can be found.
`n.clusters`	number of parallel processes to spawn.
`plot`	a suffix for the file's name containing plots of every QTL search round, e.g. "search" (default); if `NULL`, no file is produced.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.search` to be printed.
`pheno.col`	a numeric vector with the phenotype column numbers to be printed; if `NULL`, all phenotypes from `'data'` will be included.
`...`	currently ignored

Value

An object of class `qtlpoly.search` which contains a list of `results` for each trait with the following components:

`pheno.col`	a phenotype column number.
`stat`	a vector containing values from score statistics.
`pval`	a vector containing *p*-values from score statistics.
`qtls`	a data frame with information from the mapped QTL.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

Qu L, Guennel T, Marshall SL (2013) Linear score tests for variance components in linear mixed models and applications to genetic association studies. *Biometrics* 69 (4): 883–92.

Zou F, Fine JP, Hu J, Lin DY (2004) An efficient resampling method for assessing genome-wide statistical significance in mapping quantitative trait loci. *Genetics* 168 (4): 2307-16. doi:10.1534/genetics.104.031427

See Also

read_data, null_model

Examples

```
 # Estimate conditional probabilities using mappoly package
 library(mappoly)
 library(qtlpoly)
 genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
 data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

 # Build null model
 null.mod = null_model(data, pheno.col = 1, n.clusters = 1)

 # Perform forward search
search.mod = search_qtl(data, model = null.mod, w.size = 15, sig.fwd = 0.01, n.clusters = 1)
```

---

simulate_qtl *Simulations of multiple QTL*

Description

Simulate new phenotypes with a given number of QTL and creates new object with the same structure of class `qtlpoly.data` from an existing genetic map.

Usage

```
simulate_qtl(
  data,
  mu = 0,
  h2.qtl = c(0.3, 0.2, 0.1),
  var.error = 1,
  linked = FALSE,
  n.sim = 1000,
  missing = TRUE,
  w.size = 20,
  seed = 123,
  verbose = TRUE
)

## S3 method for class 'qtlpoly.simul'
print(x, detailed = FALSE, ...)
```

Arguments

`data`	an object of class `qtlpoly.data`.
`mu`	simulated phenotype mean, e.g. 0 (default).
`h2.qtl`	vector with QTL heritabilities, e.g. `c(0.3, 0.2, 0.1)` for three QTL (default); if `NULL`, only error is simulated.
`var.error`	simulated error variance, e.g. 1 (default).
`linked`	if `TRUE` (default), at least two QTL will be linked; if `FALSE`, QTL will be randomly assigned along the genetic map. Linkage is defined by a genetic distance smaller than the selected `w.size`.
`n.sim`	number of simulations, e.g. 1000 (default).
`missing`	if `TRUE` (default), phenotypes are simulated with the same number of missing data observed in `data$pheno`.
`w.size`	the window size (in centiMorgans) between two (linked) QTL, e.g. 20 (default).
`seed`	integer for the `set.seed()` function.
`verbose`	if `TRUE` (default), current progress is shown; if `FALSE`, no output is produced.
`x`	an object of class `qtlpoly.sim` to be printed.
`detailed`	if `TRUE`, detailed information on linkage groups and phenotypes in shown; if `FALSE`, no details are printed.
`...`	currently ignored

Value

An object of class `qtlpoly.sim` which contains a list of `results` with the same structure of class `qtlpoly.data`.

Author(s)

Guilherme da Silva Pereira, <`gdasilv@ncsu.edu`>

References

Pereira GS, Gemenet DC, Mollinari M, Olukolu BA, Wood JC, Mosquera V, Gruneberg WJ, Khan A, Buell CR, Yencho GC, Zeng ZB (2020) Multiple QTL mapping in autopolyploids: a random-effect model approach with application in a hexaploid sweetpotato full-sib population, *Genetics* 215 (3): 579-595. doi:10.1534/genetics.120.303080.

See Also

`read_data`

Examples

```
# Estimate conditional probabilities using mappoly package
library(mappoly)
library(qtlpoly)
genoprob4x = lapply(maps4x[c(5)], calc_genoprob)
data = read_data(ploidy = 4, geno.prob = genoprob4x, pheno = pheno4x, step = 1)

# Simulate new phenotypes
sim.dat = simulate_qtl(data = data, n.sim = 1)
sim.dat
```

Index