Package 'polyqtlR'

October 14, 2022

Type Package

Title QTL Analysis in Autopolyploid Bi-Parental F1 Populations

Version 0.0.9

Date 2022-02-02

Maintainer Peter Bourke <pbourkey@gmail.com>

Description Quantitative trait loci (QTL) analysis and exploration of meiotic patterns in
autopolyploid bi-parental F1 populations.
For all ploidy levels, identity-by-descent (IBD) probabilities can be estimated.
Significance thresholds, exploring QTL allele effects and visualising results are provided.
For more background and to reference the package see <doi:10.1093/bioinformatics/btab574>.

Depends R (>= 3.5.0)

Imports abind, doParallel, foreach, Hmisc, knitr, nlme, RColorBrewer,
Rcpp (>= 0.12.19), reshape2

Suggests igraph, mappoly (>= 0.3.0), polymapR, rmarkdown

License GPL-3

Encoding UTF-8

SystemRequirements C++11

RoxygenNote 7.1.2

VignetteBuilder knitr

LazyData true

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Peter Bourke [aut, cre],
Christine Hackett [ctb],
Chris Maliepaard [ctb],
Geert van Geest [ctb],
Roeland Voorrips [ctb],
Johan Willemsen [ctb]

Repository CRAN

Date/Publication 2022-02-02 16:40:02 UTC

R **topics documented:**

BLUE	*Calculate Best Linear Unbiased Estimates using linear mixed model from* nlme *package*

Description

Calculation of BLUEs from data frame of genotype names and phenotypes (assuming repeated measurements)

Usage

```
BLUE(data, model, random, genotype.ID)
```

Arguments

data	Data frame of genotype codes and corresponding phenotypes
model	The model specification of fixed terms, eg. Yield ~ Clones
random	The random component of the model (repeat structure, can be nested), eg. ~1 | Blocks if only Blocks are used
genotype.ID	The colname used to describe genotypes, e.g. "Clones"

Value

A data-frame with columns "geno" for the genotype names, and "blue" for the BLUEs.

Examples

```
data("Phenotypes_4x")
blue <- BLUE(data = Phenotypes_4x,model = pheno~geno,random = ~1|year,genotype.ID = "geno")
```

BLUEs.pheno	*A data-frame of best linear unbiased predicted (BLUE) phenotypes (4x)*

Description

A data-frame of best linear unbiased predicted (BLUE) phenotypes (4x)

Usage

```
BLUEs.pheno
```

Format

An object of class data.frame with 50 rows and 2 columns.

| bx | *Rcpp internal function Backward from forward-backward algorithm* |

Description

Rcpp internal function Backward from forward-backward algorithm

Usage

```
bx
```

Format

An object of class `function` of length 1.

| check_cofactors | *Build a multi-QTL model using step-wise procedure of checking genetic co-factors.* |

Description

The function `check_cofactors` initially fits all significant QTL positions as co-factors, both individually and in combination. Significance thresholds are re-estimated each time, yielding threshold-corrected LOD scores. If this leads to a change in the estimated position of QTL, or detection of subsequent peaks, a second round of co-factor inclusion is performed for all new QTL or novel QTL combinations. Finally, the multi-QTL model that maximises the individual significance of each QTL is returned as a data.frame. This can be directly passed to the function PVE to estimate the percentage variance explained by the full multi-QTL model and all possible sub-models. Note: this function estimates the most likely QTL positions by maximising the threshold-corrected LOD at QTL peaks. Non-additive interactions between QTL may be missed as a result. It is recommended to run a manual co-factor analysis as well, as described in the package vignette.

Usage

```
check_cofactors(
  IBD_list,
  Phenotype.df,
  genotype.ID,
  trait.ID,
  LOD_data,
  min_res = 20,
  ncores = 1,
  verbose = TRUE
)
```

Arguments

IBD_list	List of IBD_probabilities as estimated using one of the various methods available (e.g. estimate_IBD).
Phenotype.df	A data.frame containing phenotypic values
genotype.ID	The colname of Phenotype.df that contains the population identifiers (F1 names) (must be a colname of Phenotype.df)
trait.ID	The colname of Phenotype.df that contains the response variable to use in the model (must be a colname of Phenotype.df)
LOD_data	Output of QTLscan function.
min_res	The minimum genetic distance (resolution) assumed possible to consider 2 linked QTL (on the same linkage group) as independent. By default a value of 20 cM is used. This is not to suggest that 20 cM is a realistic resolution in a practical mapping study, but it provides the function with a criterion to consider 2 significant QTL within this distance as one and the same. For this purpose, 20 cM seems a reasonable value to use. In practice, closely linked QTL will generally "explain" all the variation at nearby positions, making it unlikely to be able to disentangle their effects. QTL positions will vary slightly when co-factors are introduced, but again this variation is presumed not to exceed 20 cM either side.
ncores	How many CPU cores should be used in the evaluation? By default 1 core is used.
verbose	Logical, by default TRUE - should progress messages be printed to the console?

Value

Data frame with the following columns:

- LG: Linkage group identifier

- cM: CentiMorgan position

- deltaLOD: The difference between the LOD score at the peak and the significance threshold (always positive, otherwise the QTL would not be significant)

- CofactorID: A identifier giving the co-factor model used in detecting the QTL (if no co-factors were included then NA). The co-factor model is described by concatenating all co-factor positions with a '+', so for example 1_10+4_20 would mean a co-factor model with 2 positions included as co-factors, namely 10 cM on linkage group 1 and 20 cM on linkage group 4.

Examples

```
data("IBD_4x","BLUEs.pheno","qtl_LODs.4x")
check_cofactors(IBD_list=IBD_4x,Phenotype.df=BLUEs.pheno,
genotype.ID="Geno",trait.ID="BLUE",LOD_data=qtl_LODs.4x)
```

---

convert_mappoly_to_phased.maplist

Function to extract the phased map from a mappoly.map object

Description

Convert MAPpoly.map object into a phased maplist, needed for IBD estimation

Usage

```
convert_mappoly_to_phased.maplist(mappoly_object)
```

Arguments

mappoly_object   An object of class 'mappoly.map', for example output of the function mappoly::est_rf_hmm_sequentia

Value

A phased.maplist, with linkage group names LG1 etc. Each list item is a data.frame with columns marker, position followed by the phased map, coded in 1 and 0 for presence/absence of SNP (alternative) allele on parental homologues (h) numbered 1:ploidy for parent 1 and ploidy + 1 : 2*ploidy for parent 2.

Examples

```
## Not run:
library("mappoly")
phased.maplist <- convert_mappoly_to_phased.maplist(maps.hexafake)

## End(Not run)
```

---

count_recombinations   *Predict recombination breakpoints using IBD probabilities*

Description

The function count_recombinations returns a list of all predicted recombination breakpoints. The output can be passed using the argument recombination_data to the function visualiseHaplo, where the predicted breakpoints overlay the haplotypes. Alternatively, a genome-wide visualisation of the recombination landscape both per linkage group and per individual can be generated using the function plotRecLS, which can be useful in identifying problematic areas of the linkage maps, or problematic individuals in the population. Currently, recombination break-points are only estimated from bivalents in meiosis; any offspring resulting from a predicted multivalent is excluded from the analysis and will be returned with a NA value.

Usage

```
count_recombinations(IBD_list, plausible_pairing_prob = 0.4)
```

Arguments

IBD_list   List of IBD_probabilities as estimated using one of the various methods available (e.g. estimate_IBD).

plausible_pairing_prob

The minimum probability of a pairing configuration needed to analyse an individual's IBD data. The default setting of 0.4 is rather low - but accommodates scenarios where e.g. two competing plausible pairing scenarios are possible. In such situations, both pairing configurations (also termed "valencies") would be expected to have a probability close to 0.5. Both are then considered, and the output contains the probability of both situations. These can then be used to generate a probabilistic recombination landscape. If a more definite set of predictions is required, simply increase plausible_pairing_prob to eliminate such uncertainty. These individuals will then be returned with a NA value.

Value

A nested list corresponding to each linkage group. Within each LG, a list with 3 items is returned, specifying the plausible_pairing_prob, the map and the predicted recombinations in each individual in the mapping population. Per individual, all valencies with a probability greater than plausible_pairing_prob are returned, specifying both the Valent_probability and the best estimate of the cM position of the recombination_breakpoints involving pairs of homologues A, B, C etc. (in the order parent 1, parent 2). If no recombinations are predicted, a NA value is given instead.

Examples

```
data("IBD_4x")
recom.ls <- count_recombinations(IBD_4x)
```

---

estimate_GIC   *Estimate the Genotypic Information Coefficient (GIC)*

Description

Function to estimate the GIC per homologue using IBD probabilities

Usage

```
estimate_GIC(IBD_list)
```

Arguments

IBD_list   List of IBD probabilities

Value

A nested list; each list element (per linkage group) contains the following items:

- GIC : Matrix of GIC values estimated from the IBD probabilities
- map : Integrated linkage map positions of markers used in IBD calculation
- parental_phase : The parental marker phasing, coded in 1 and 0's

Examples

```
data("IBD_4x")
GIC_4x <- estimate_GIC(IBD_list = IBD_4x)
```

---

estimate_IBD *Generate IBD probabilities from marker genotypes and a phased linkage map*

Description

`estimate_IBD` is a function for creating identity-by-descent (IBD) probabilities. Two computational methods are offered: by default IBD probabilites are estimated using hidden Markov models, but a heuristic method based on Bourke et al. (2014) is also included. Basic input data for this function are marker genotypes (either discrete marker dosages (ie scores 0, 1, ..., ploidy representing the number of copies of the marker allele), or the probabilities of these dosages) and a phased linkage map. Details on each of the methods are included under `method`

Usage

```
estimate_IBD(
  input_type = "discrete",
  genotypes,
  phased_maplist,
  method = "hmm",
  remove_markers = NULL,
  ploidy,
  ploidy2 = NULL,
  parent1 = "P1",
  parent2 = "P2",
  individuals = "all",
  log = NULL,
  map_function = "haldane",
  bivalent_decoding = TRUE,
  error = 0.01,
  full_multivalent_hexa = FALSE,
  verbose = FALSE,
  ncores = 1,
  fix_threshold = 0.1,
  factor_dist = 1
)
```

Arguments

input_type	Can be either one of 'discrete' or 'probabilistic'. For the former (default), dosage_matrix must be supplied, while for the latter probgeno_df must be supplied. Note that probabilistic genotypes can only be accepted if the method is default ('hmm').
genotypes	Marker genotypes, either a 2d matrix of integer marker scores or a data.frame of dosage probabilities. Details are as follows: • discrete : If input_type is 'discrete', genotypes is a matrix of marker dosage scores with markers in rows and individuals in columns. Both (marker) rownames and (individual or sample) colnames are needed. • probabilistic : If input_type is 'probabilistic', genotypes is a data frame as read from the scores file produced by function saveMarkerModels of R package fitPoly, or alternatively, a data frame containing at least the following columns: – SampleName : Name of the sample (individual) – MarkerName : Name of the marker – P0 : Probabilities of dosage score '0' – P1, P2... etc. : Probabilities of dosage score '1' etc. (up to max offspring dosage, e.g. P4 for tetraploid population)
phased_maplist	A list of phased linkage maps, the output of polymapR::create_phased_maplist
method	The method used to estimate IBD probabilities, either "hmm" or "heur". By default, the Hidden Markov Model (hmm) method is used. This uses an approach developed by Zheng et al (2016), and implemented in the 'TetraOrigin' package. However, unlike the original TetraOrigin software, it does not re-estimate parental linkage phase, as this is assumed to have been generated during map construction. Alternatively, a heuristic algorithm can be employed (method = "heur"), providing computational efficiency at higher ploidy levels (hexaploid, octoploid etc.), but at the cost of some accuracy. If method = "hmm" is specified, only diploid, triploid, autotetraploid and autohexaploid populations are currently allowed, while method = "heur" caters for all possible ploidy levels. Furthermore, the argument bivalent_decoding can only be set to FALSE in the case of the 'hmm' method (i.e. allowing for the possibility of multivalent formation and double reduction).
remove_markers	Optional vector of marker names to remove from the maps. Default is NULL.
ploidy	Integer. Ploidy of the organism.
ploidy2	Optional integer, by default NULL. Ploidy of parent 2, if different from parent 1.
parent1	Identifier of parent 1, by default assumed to be "P1"
parent2	Identifier of parent 2, by default assumed to be "P2"
individuals	By default "all" offspring are included, but otherwise a subset can be selected, using a vector of offspring indexing numbers (1,2, etc.) according to their order in dosage_matrix
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

map_function	Mapping function to use when converting map distances to recombination frequencies. Currently only "haldane" or "kosambi" are allowed.
bivalent_decoding	
	Option to consider only bivalent pairing during formation of gametes (ignored for diploid populations, as only bivalents possible there), by default TRUE
error	The (prior) probability of errors in the offspring dosages, usually assumed to be small but non-zero
full_multivalent_hexa	
	Option to allow multivalent pairing in both parents at the hexaploid level, by default FALSE. Note that if TRUE, a very large available RAM may be required (>= 32Gb) to process the data.
verbose	Logical, by default TRUE. Should progress messages be written?
ncores	How many CPU cores should be used in the evaluation? By default 1 core is used.
fix_threshold	If method = "heur", the threshold to fix the IBD probabilities while correcting for the sum of probabilities.
factor_dist	If method = "heur", the factor by which to increase or decrease the recombination frequencies as calculated from the map distances.

Value

A list of IBD probabilities, organised by linkage group (as given in the input phased_maplist). Each list item is itself a list containing the following:

- IBDtype: The type of IBD; for this function only "genotypeIBD" are calculated.
- IBDarray: A 3d array of IBD probabilities, with dimensions marker, genotype-class and F1 individual.
- map: A 3-column data-frame specifying chromosome, marker and position (in cM)
- parental_phase: Phasing of the markers in the parents, as given in the input phased_maplist
- marginal.likelihoods: A list of marginal likelihoods of different valencies if method "hmm" was used, otherwise NULL
- valency: The predicted valency that maximised the marginal likelihood, per offspring. For method "heur", NULL
- offspring: Offspring names
- biv_dec: Logical, whether bivalent decoding was used in the estimation of the F1 IBD probabilities.
- gap: The size of the gap (in cM) used when interpolating the IBD probabilities. See function [spline_IBD](#) for details.
- genocodes: Ordered list of genotype codes used to represent different genotype classes.
- pairing: log likelihoods of each of the different pairing scenarios considered (can be used e.g. for post-mapping check of preferential pairing)
- ploidy: ploidy of parent 1
- ploidy2: ploidy of parent 2
- method: The method used, either "hmm" (default) or "heur". See argument method
- error: The error prior used, if method "hmm" was used, otherwise NULL

References

- Durbin R, Eddy S, Krogh A, Mitchison G (1998) Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge: Cambridge University Press.

- Hackett et al. (2013) Linkage analysis and QTL mapping using SNP dosage data in a tetraploid potato mapping population. PLoS One 8(5): e63939

- Zheng et al. (2016) Probabilistic multilocus haplotype reconstruction in outcrossing tetraploids. Genetics 203: 119-131

- Bourke P.M. (2014) QTL analysis in polyploids: Model testing and power calculations. Wageningen University (MSc thesis)

Examples

```
data("phased_maplist.4x", "SNP_dosages.4x")
estimate_IBD(phased_maplist=phased_maplist.4x,genotypes=SNP_dosages.4x,ploidy=4)
```

---

exploreQTL	*Explore the possible segregation type of a QTL peak using Schwarz Information Criterion*

Description

Function to explore the possible segregation type at a QTL position using the Schwarz Information Criterion

Usage

```
exploreQTL(
  IBD_list,
  Phenotype.df,
  genotype.ID,
  trait.ID,
  linkage_group,
  LOD_data,
  cM = NULL,
  QTLconfig = NULL,
  plotBIC = TRUE,
  deltaBIC = 6,
  testAllele_Effects = TRUE,
  log = NULL
)
```

Arguments

IBD_list	List of IBD probabilities
Phenotype.df	A data.frame containing phenotypic values

genotype.ID	The colname of `Phenotype.df` that contains the population identifiers (F1 names) (must be a colname of `Phenotype.df`)
trait.ID	The colname of `Phenotype.df` that contains the response variable to use in the model (must be a colname of `Phenotype.df`)
linkage_group	Numeric identifier of the linkage group being tested, based on the order of `IBD_list`. Only a single linkage group is allowed.
LOD_data	Output of [QTLscan](#) function
cM	By default `NULL`, in which case the position of maximum LOD score is taken as the position of interest. Otherwise, the cM position to be explored.
QTLconfig	Nested list of homologue configurations and modes of action of QTL to be explored and compared, the output of [segMaker](#). Note that a default List is available of all possible bi-allelic QTL if none is provided. Each list element is itself a list with components

> - homs : a vector of length at least 1, describing the proposed homologues the functional allele Q is on
> - mode : Vector of same length as homs with codes "a" for additive and "d" for dominant.

plotBIC	Logical, with default `TRUE` - should the calculated BIC values be plotted?
deltaBIC	Numeric, by default 6. Configurations within this distance of the minimum BIC are considered plausible.
testAllele_Effects	
	Logical, with default `TRUE` - should the effects of the different alleles be tested using the most likely QTL configuration?
log	Character string specifying the log filename to which standard output should be written. If `NULL` log is send to stdout.

Value

List with the following items:

- BIC : Vector of BIC values corresponding to elements of `QTLconfig` provided for testing
- Allele.effects : Summary of the means and standard errors of groups with (+) and without(-) the specified allele combinations for the most likely QTLconfig if `testAllele_Effects = TRUE` (NULL otherwise).

Examples

```
data("IBD_4x","BLUEs.pheno","qtl_LODs.4x")
exploreQTL(IBD_list = IBD_4x,
          Phenotype.df = BLUEs.pheno,
          genotype.ID = "Geno",
          trait.ID = "BLUE",
          linkage_group = 1,
          LOD_data = qtl_LODs.4x)
```

---

findPeak   *Function to find the position of maximum LOD on a particular linkage group*

Description

Given QTL output, this function returns the position of maximum LOD for a specified linkage group.

Usage

```
findPeak(LOD_data, linkage_group, verbose = TRUE)
```

Arguments

LOD_data   Output of [QTLscan](#) function.

linkage_group   Numeric identifier of the linkage group being tested, based on the order of IBD_list. Only a single linkage group is allowed.

verbose   Should messages be written to standard output? By default TRUE.

Examples

```
data("qtl_LODs.4x")
findPeak(LOD_data=qtl_LODs.4x,linkage_group=1)
```

---

findSupport   *Function to find a LOD - x support interval around a QTL position*

Description

Given QTL output, this function returns the LOD - x support for a specified linkage group, taking the maximum LOD position as the desired QTL peak.

Usage

```
findSupport(LOD_data, linkage_group, LOD_support = 2)
```

Arguments

LOD_data   Output of [QTLscan](#) function.

linkage_group   Numeric identifier of the linkage group being tested, based on the order of IBD_list. Only a single linkage group is allowed.

LOD_support   The level of support around a QTL peak, by default 2 (giving a LOD - 2 support interval, the range of positions with a LOD score within 2 LOD units of the maximum LOD on that linkage group).

Examples

```
data("qtl_LODs.4x")
findSupport(LOD_data=qtl_LODs.4x,linkage_group=1)
```

---

fx   *Rcpp internal function Forward from forward-backward algorithm*

Description

Rcpp internal function Forward from forward-backward algorithm

Usage

```
fx
```

Format

An object of class function of length 1.

GIC_4x   *A list of GIC estimates (4x)*

Description

A list of GIC estimates (4x)

Usage

```
GIC_4x
```

Format

An object of class list of length 2.

---

hexa.list   *A list of hexaploid bivalent pairing configurations*

Description

A list of hexaploid bivalent pairing configurations

Usage

```
hexa.list
```

Format

An object of class list of length 15.

IBD_4x   *A list of identity-by-descent probabilities (4x)*

Description

A list of identity-by-descent probabilities (4x)

Usage

```
IBD_4x
```

Format

An object of class `list` of length 2.

---

import_IBD   *Import IBD probabilities as estimated by TetraOrigin*

Description

Imports the summarised IBD probability output of TetraOrigin (which estimates IBD probabilities at all marker positions), and interpolates these at a grid of positions at user-defined spacing.

Usage

```
import_IBD(
  folder = NULL,
  filename.vec,
  bivalent_decoding = TRUE,
  error = 0.01,
  log = NULL
)
```

Arguments

folder	The path to the folder in which the TetraOrigin output is contained, default is `NULL` if files are in working directory.
filename.vec	A vector of the character filename(s) of the `.csv` file(s) containing the output of TetraOrigin. Should be in order according to LG/chromosome numbering.
bivalent_decoding	
	Logical, if `TRUE` then only bivalent pairing was allowed in TetraOrigin, specify `FALSE` if multivalent pairing was also allowed.
error	The offspring error prior used in the offspring decoding step, here assumed to be 0.01
log	Character string specifying the log filename to which standard output should be written. If `NULL` log is send to stdout.

Value

Returns a list with the following items:

IBDtype :   Always "genotypeIBD" for the output of TetraOrigin

IBDarray :   An array of IBD probabilities. The dimensions of the array are: markers, geno-
                    type classes and individuals.

map :   Integrated linkage map positions of markers used in IBD calculation

parental_phase :
                    The parental marker phasing as used by TetraOrigin, recoded in 1 and 0's

marginal.likelihoods :
                    A list of marginal likelihoods of different valencies, currently NULL

valency :   The predicted valency that maximised the marginal likelihood, per offspring.
                    Currently NULL

offspring :   Offspring names

biv_dec :   Logical, the bivalent_decoding parameter specified.

gap :   The gap size used in IBD interpolation, by default NULL. See spline_IBD

genocodes :   Ordered list of genotype codes used to represent different genotype classes.

pairing :   log likelihoods of each of the different pairing scenarios considered (can be used
                    e.g. for post-mapping check of preferential pairing)

ploidy :   The ploidy of parent 1, by default assumed to be 4

ploidy2 :   The ploidy of parent 2, by default assumed to be 4

method :   The method used, always returned as "hmm_TO" (Hidden Markov Model TetraO-
                    rigin)

error :   The error prior used in the calculation, assumed to be 0.01

---

impute_dosages   *Re-estimate marker dosages given IBD input estimated using a high*
                          error prior.

Description

Function to correct marker dosage scores given a list of previously estimated IBD probabilities. This
may prove useful to correct genotyping errors. Running the estimate_IBD function with a high
error prior will result in suppressed predictions of double recombination events, associated with
genotyping errors. By forcing the HMM to penalise double recombinations heavily, a smoothed
haplotype landscape is achieved in which individual genotype observations are down-weighted.
This smoothed output is then used to re-estimate marker dosages, dependent on (correct) parental
scores. An alternative strategy is to use the function maxL_IBD over a range of error priors first, and
use the resulting $maxL_IBD output as input here (as the IBD_list). In this case, set the argument
min_error_prior to a low value (0.005 say) to avoid issues.

Usage

```
impute_dosages(
  IBD_list,
  dosage_matrix,
  parent1 = "P1",
  parent2 = "P2",
  rounding_error = 0.05,
  min_error_prior = 0.1,
  verbose = TRUE
)
```

Arguments

`IBD_list`	List of IBD probabilities
`dosage_matrix`	An integer matrix with markers in rows and individuals in columns. Note that probabilistic genotypes are not currently catered for here.
`parent1`	The identifier of parent 1, by default "P1"
`parent2`	The identifier of parent 2, by default "P2"
`rounding_error`	The maximum deviation from an integer value that an inputed value can have, by default 0.05. For example, an imputed score of 2.97 or 3.01 would both be rounded to a dosage of 3, while 2.87 would be deemed too far from an integer score, and would be made missing. If you find the output contains too many missing values, a possibility would be to increase the `rounding_error`. However this may also introduce more errors in the output!
`min_error_prior`	
	Suggestion for a suitably high error prior to be used in IBD calculations to ensure IBD smoothing is achieved. If IBD probabilities were estimated with a smaller error prior, the function aborts.
`verbose`	Should messages be written to standard output?

Examples

```
## Not run:
# Toy example only, as this will result in an Error: the original error prior was too low
data("IBD_4x","SNP_dosages.4x")
impute_dosages(IBD_list=IBD_4x,dosage_matrix=SNP_dosages.4x)

## End(Not run)
```

---

maxL_IBD	*Wrapper function to run estimate_IBD function over multiple error priors*

Description

Function to run the `estimate_IBD` function over a range of possible error priors. The function returns a merged set of results that maximise the marginal likelihood per individual, i.e. allowing a per-individual error rate within the options provided in the errors argument.

Usage

```
maxL_IBD(errors = c(0.01, 0.05, 0.1, 0.2), ...)
```

Arguments

errors	Vector of offspring error priors to test (each between 0 and 1)
...	Arguments passed to `estimate_IBD`.

Value

A list containing the following components:

- maxL_IBD : A nested list as would have been returned by the estimate_IBD function, but composite across error priors to maximise the marginal likelihoods. Note that the $error values per linkage group are now the average error prior across the population per linkage group

- MML : A 3d array of the maximal marginal likelihoods, per error prior. Dimensions are individuals, linkage groups, error priors.

- error_per_ind : A matrix of the most likely genotyping error rates per individual (in rows) for each linkage group (in columns)

- errors : The error priors used (i.e. the input vector is returned for later reference.)

Examples

```
## Not run:
data("phased_maplist.4x","SNP_dosages.4x")
maxL_IBD(phased_maplist=phased_maplist.4x,genotypes=SNP_dosages.4x,
ploidy=4,errors=c(0.01,0.02,0.05,0.1))

## End(Not run)
```

---

meiosis_report	*Generate a 'report' of predicted meiotic behaviour in an F1 population*

Description

Function to extract the chromosome pairing predictions as estimated by `estimate_IBD`. Apart from producing an overview of the pairing during parental meiosis (including counts of multivalents, per linkage group per parent), the function also applies a simple chi-squared test to look for evidence of non-random pairing behaviour from the bivalent counts (deviations from a polysomic model)

Usage

```
meiosis_report(
  IBD_list,
  visualise = FALSE,
  plausible_pairing_prob = 0.9,
  precision = 2
)
```

Arguments

IBD_list   List of IBD probabilities as estimated by `estimate_IBD` using method 'hmm',
                 or externally (e.g. using TetraOrigin)

visualise   Logical, by default `FALSE`. If `TRUE`, a plot of the pairing results is produced
                 per LG. In order to flag extreme deviations from the expected numbers (associ-
                 ated with polysomic inheritance, considered the Null hypothesis), barplots are
                 coloured according to the level of significance of the X2 test. Plots showing red
                 bars indicate extreme deviations from a polysomic pattern.

plausible_pairing_prob

                 The minimum probability of a pairing configuration needed to analyse an indi-
                 vidual's IBD data.

precision   To how many decimal places should summed probabilities per bivalent pairing
                 be rounded? By default 2.

Value

The function returns a nested list, with one element per linkage group in the same order as the input
IBD list. Per linkage group, a list is returned containing the following components:

- P1_multivalents: The count of multivalents in parent 1 (only relevant if `bivalent_decoding` = FALSE during IBD calculation)
- P2_multivalents: Similarly, the count of multivalents in parent 2
- P1_pairing: The counts of each bivalent pairing predicted in parent 1, with an extra column Pr(X2) which gives the p-value of the X2 test of the off-diagonal terms in the matrix. In the case of a tetraploid, pairing A with B automatically implies C with D pairing, so the count table contains a lot of redundancy. The table should be read using both row and column names, so row A and column B corresponds to the count of individuals with A and B pairing (and hence C and D pairing). In a hexaploid, A-B pairing does not imply a particular pairing configuration in the remaining homologues. In this case, row A and column B is the count of individuals where A and B were predicted to have paired, summed over all three bivalent configurations with A and B paired (AB-CD-EF, AB-CE-DF, AB-CF,DE).
- P2_pairing: Same as P1_pairing, except using parent 2
- ploidy: The ploidy of parent 1
- ploidy2: The ploidy of parent 2

Examples

```
data("IBD_4x")
mr.ls<-meiosis_report(IBD_list = IBD_4x)
```

---

mr.ls   *A list of pairing predictions (4x)*

Description

A list of pairing predictions (4x)

Usage

```
mr.ls
```

Format

An object of class `list` of length 2.

---

phased_maplist.4x   *A list of phased maps (4x)*

Description

A list of phased maps (4x)

Usage

```
phased_maplist.4x
```

Format

An object of class `list` of length 2.

Phenotypes_4x   *A data-frame of phenotypes (4x)*

Description

A data-frame of phenotypes (4x)

Usage

```
Phenotypes_4x
```

Format

An object of class `data.frame` with 150 rows and 3 columns.

---

plotLinearQTL *Plot the results of genome-wide QTL analysis along a single track*

Description

QTL plotting function that plots output of `QTLscan` function along a single track, useful for overlaying plots. Only works for scan over multiple chromosomes.

Usage

```
plotLinearQTL(
  LOD_data,
  inter_chm_gap = 5,
  overlay = FALSE,
  ylimits = NULL,
  sig.unit = "LOD",
  plot_type = c("lines", "points"),
  add_xaxis = TRUE,
  add_rug = TRUE,
  add_thresh = TRUE,
  override_thresh = NULL,
  thresh.lty = 3,
  thresh.lwd = 2,
  thresh.col = "darkred",
  return_plotData = FALSE,
  show_thresh_CI = TRUE,
  use_LG_names = TRUE,
  axis_label.cex = 1,
  custom_LG_names = NULL,
  LGdiv.col = "gray42",
  ...
)
```

Arguments

LOD_data	Output of `QTLscan` function.
inter_chm_gap	The gap size (in cM) between successive chromosomes - by default a gap of 5 cM is used.
overlay	Add to an existing plot (should be produced by a comparable call to this function) or not? By default `FALSE`, in which case a new plot is drawn. Can be useful for displaying results of multiple analyses together. However, an alternative approach, when significance thresholds have been calculated for multiple comparable scans, that plots be rescaled so that significance thresholds overlap perfectly. For this, the `plotLinearQTL_list` function is advised.
ylimits	Use to specify ylimits of plot region, though by default `NULL` in which case a suitable plot region is automatically used.

sig.unit	Label to use on the y-axis for significance units, by default assumed to be LOD score.
plot_type	Plots can be either in line drawings ("lines") or scatter plot format ("points").
add_xaxis	Should an x-axis be drawn? If multiple QTL analyses are performed on different traits, specifying this to be FALSE and using par(mar=c(0,4.1,4.1,2.1)) allows subsequent plots to be neatly stacked.
add_rug	Logical, by default TRUE - should original marker points be added to plot?
add_thresh	Logical, by default TRUE - should a significance threshold be added to plot?
override_thresh	By default NULL. Can be used to specify a value for the significance threshold, overriding any stored in LOD_data.
thresh.lty	Gives user control over the line type of the significance threshold to be drawn.
thresh.lwd	Gives user control over the line width of the significance threshold to be drawn.
thresh.col	Gives user control over the line colour of the significance threshold to be drawn.
return_plotData	Logical, by default FALSE. If TRUE, then the x and y coordinates of the plot data are returned, which can be useful for subsequent plot manipulations and overlays.
show_thresh_CI	Logical, by default TRUE. Should confidence interval bounds around LOD threshold be shown?
use_LG_names	Logical, by default TRUE. Should original character LG names be used as axis labels, or should numbering be used instead?
axis_label.cex	Argument to adjust the size of the axis labels, can be useful if there are many linkage groups to plot
custom_LG_names	Specify a vector that contains custom linkage group names. By default NULL
LGdiv.col	Colour of dividing lines between linkage groups, by default grey.
...	Arguments passed to [plot](), and [lines]() or [points]() as appropriate (see argument plot_type).

Value

The plot data, if return_plotData = TRUE. Otherwise NULL

Examples

```
## Not run:
data("qtl_LODs.4x")
plotLinearQTL(LOD_data = qtl_LODs.4x)

## End(Not run)
```

---

plotLinearQTL_list	*Overlay the results of a number of genome-wide QTL analysis for which significance thresholds are available.*

Description

Extension of the [plotLinearQTL](#) function, taking as input a list generated from combining the output of [QTLscan](#). Its distinguishing characteristic is that overlaid plots are re-scaled so that the significance thresholds overlap. This can be useful if there are multiple results being plotted together for comparison, all of which may have different thresholds. The resulting plot can help quickly compare the power of different analyses. Warning - the y axis LOD scale is only correct for the first list element / set of results. Also as before, this function only works for QTL scan over multiple chromosomes.

Usage

```
plotLinearQTL_list(
  LOD_data.ls,
  inter_chm_gap = 5,
  ylimits = NULL,
  sig.unit = "LOD",
  plot_type,
  add_xaxis = TRUE,
  add_rug = TRUE,
  colours = c("black", "red", "dodgerblue", "sienna4"),
  ylab.at = 2.5,
  main.size = 2,
  main.lty = 1,
  thresh.lty = 3,
  thresh.lwd = 2,
  thresh.col = "darkred",
  return_plotData = FALSE,
  highlight_positions = NULL,
  LGdiv.col = "gray42",
  use_LG_names = TRUE,
  ...
)
```

Arguments

LOD_data.ls	A list, each element of which is a separate output of [QTLscan](#), for which the setting perm_test = TRUE was used each time.
inter_chm_gap	The gap size (in cM) between successive chromosomes - by default a gap of 5 cM is used.
ylimits	Use to specify ylimits of plot region, though by default NULL in which case a suitable plot region is automatically used.

sig.unit	Label to use on the y-axis for significance units, by default assumed to be "LOD".
plot_type	Plots can be either in line drawings or scatter plot format. If multiple types are required, supply as a vector of same length as LOD_data.ls
add_xaxis	Should an x-axis be drawn? If multiple QTL analyses are performed on different traits, specifying this to be FALSE and using par(mar=c(0,4.1,4.1,2.1)) allows subsequent plots to be neatly stacked.
add_rug	Logical, by default TRUE - should original marker points be added to plot?
colours	Vector of colours to be used in the plotting. A default set of 4 colours is provided.
ylab.at	Distance from the y-axis to place label (by default at 2.5 points)
main.size	Size of line (or point) to plot the "main" data, the first set of results in the LOD_data.ls input list, by default 2.
main.lty	Line type for the "main" data, by default a normal line (lty = 1).
thresh.lty	Gives user control over the line type of the significance threshold to be drawn.
thresh.lwd	Gives user control over the line width of the significance threshold to be drawn.
thresh.col	Gives user control over the line colour of the significance threshold to be drawn, by default "darkred"
return_plotData	
	Logical, by default FALSE. If TRUE, then the x and y coordinates of the plot data are returned, which can be useful for subsequent plot manipulations and overlays.
highlight_positions	
	Option to include a list of associated positions to highlight, of the same length and in the same order as LOD_data.ls. Each set of positions should be provided in data.frame format with 3 columns corresponding to linkage group, type and ID (same format as the cofactor.df argument of function [QTLscan](). This can be useful if genetic co-factor analyses are being compared. If no position is to be highlighted, add the corresponding list element as NULL.
LGdiv.col	Colour of dividing lines between linkage groups, by default grey.
use_LG_names	Logical, by default TRUE. Should original character LG names be used as axis labels, or should numbering be used instead?
...	Arguments passed to [lines]() or [points]() as appropriate (see argument plot_type).

Value

The plot data, if return_plotData = TRUE, otherwise NULL

Examples

```
## Not run:
data("qtl_LODs.4x")
## Introduce some arbitrary noise for the sake of this example:
qtl_LODs.4x_2 <- qtl_LODs.4x
qtl_LODs.4x_2$Perm.res$threshold <- 2.5
qtl_LODs.4x_2$QTL.res$LOD<-qtl_LODs.4x_2$QTL.res$LOD+rnorm(length(qtl_LODs.4x_2$QTL.res$LOD),2)
```

```
plotLinearQTL_list(list(qtl_LODs.4x,qtl_LODs.4x_2),plot_type="lines")

## End(Not run)
```

---

plotQTL   *Plot the results of a previous QTL analysis*

Description

Basic QTL plotting function, taking map positions and significance levels as input

Usage

```
plotQTL(
  LOD_data,
  support_interval = 0,
  ylimits = NULL,
  multiplot = NULL,
  plot_type = "lines",
  overlay = FALSE,
  add_xaxis = TRUE,
  add_rug = TRUE,
  mainTitle = FALSE,
  log = NULL,
  ...
)
```

Arguments

LOD_data   Output list from [QTLscan](#) with items QTL.res and Perm.res (the latter can be NULL)

support_interval
                  Numeric. If 0 (by default) then there is no support interval returned. If greater than zero, then a LOD support interval is shown on output plot and the bounds are returned.

ylimits   Use to specify ylimits of plot region, though by default NULL in which case a suitable plot region is automatically used.

multiplot   Vector of integers. By default NULL. If LOD_data contains results from multiple linkage groups, you can define the number of rows and columns in the plot layout.

plot_type   How should be plots be drawn, either "lines" or "points" are possible

overlay   Add to an existing plot (should be produced by a comparable call to this function) or not? By default FALSE, in which case a new plot is drawn. Can be useful for displaying results of multiple analyses together.

add_xaxis	Should an x-axis be drawn? If multiple QTL analyses are performed on different traits, specifying this to be FALSE and using par(mar=c(0,4.1,4.1,2.1)) allows subsequent plots to be neatly stacked.
add_rug	Logical, by default TRUE - should original marker points be added to plot?
mainTitle	Vector of plot titles (single character vector also allowed and will be recycled). For no plot titles, leave as FALSE
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
...	Extra arguments passed to plotting functions (plot, lines / points)

Value

The cM bounds of the LOD support interval, if support_interval > 0.

Examples

```
data("qtl_LODs.4x")
plotQTL(LOD_data = qtl_LODs.4x,multiplot = c(1,2),ylimits = c(0,5), plot_type = "points")
```

---

plotRecLS   *Plot the recombination landscape across the genome*

Description

Function which visualises the recombination landscape in two ways: per linkage group, and per individual. For the first analysis, a rudimentary spline is also fitted to estimate the recombination rate along a grid of positions defined by gap, which is also returned by the function.

Usage

```
plotRecLS(
  recombination_data,
  plot_per_LG = TRUE,
  plot_per_ind = TRUE,
  gap = 1,
  ...
)
```

Arguments

recombination_data

                Data on predicted recombination events, as returned by the function [count_recombinations](#)

plot_per_LG	Logical argument, plot recombination events per linkage group? By default TRUE.
plot_per_ind	Logical argument, plot recombination events per individual? By default TRUE.

gap	The size (in cM) of the gap used to define the grid of positions to define the window in which to estimate recombination rate. By default 1 cM. Interpolated positions are taken to be the centre of an interval, so a 1 cM gap would result in predictions for positions 0.5 cM, 1.5 cM etc.
...	Option to pass extra arguments to the [plot](#) function for the per_LG plots. This may lead to conflicts with arguments already declared internally (such as main for example).

Value

A list with two elements, per_LG and per_individual. The first of these is itself a list with the same length as recombination_data, giving the estimated recombination rates along the linkage group. This rate is simply estimated as the (weighted) count of recombination breakpoints divided by the population size.

Examples

```
data("Rec_Data_4x")
plotRecLS(Rec_Data_4x)
```

---

polyqtlR	*QTL analysis in polyploid species using identity-by-descent probabilities*

Description

R package to perform QTL analysis using marker data from polyploid species.

PVE	*Function to determine the percentage variance explained (PVE) of a (maximal) QTL model, and explore sub-models.*

Description

This function builds a (maximal) QTL model from previously detected QTL peaks and outputs the percentage variance explained (PVE) of the full QTL model and all sub-models. It uses a similar approach to the fitting of genetic co-factors in the function [QTLscan](#). The PVE is very similar to but not exactly equal to the adjusted R2 returned in QTLscan at each position (and note: in the former case, these R2 values are per-locus, while this function can estimate the PVE combined over multiple loci). The discrepancy has to do with how PVE is calculated using the formula $100(1 - RSS0/RSS1)$, where RSS0 and RSS1 are the residual sums of squares of the NULL and QTL models, respectively.

Usage

```
PVE(
  IBD_list,
  Phenotype.df,
  genotype.ID,
  trait.ID,
  block = NULL,
  QTL_df = NULL,
  prop_Pheno_rep = 0.5,
  log = NULL,
  verbose = FALSE
)
```

Arguments

`IBD_list`	List of IBD probabilities
`Phenotype.df`	A data.frame containing phenotypic values
`genotype.ID`	The colname of `Phenotype.df` that contains the offspring identifiers (F1 names)
`trait.ID`	The colname of `Phenotype.df` that contains the response variable to use in the model
`block`	The blocking factor to be used, if any (must be colname of `Phenotype.df`). By default NULL, in which case no blocking structure (for unreplicated experiments)
`QTL_df`	A 2-column data frame of previously-detected QTL; column 1 gives linkage group identifiers, column 2 specifies the cM position of the QTL. If not specified, an error results. It can be convenient to generate a compatible data.frame by first running the function `check_cofactors` to build a multi-QTL model.
`prop_Pheno_rep`	The minimum proportion of phenotypes represented across blocks. If less than this, the individual is removed from the analysis. If there is incomplete data, the missing phenotypes are imputed using the mean values across the recorded observations.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
`verbose`	Should messages be written to standard output?

Value

A list with percentage variance explained of maximal QTL model and all sub-models

Examples

```
data("IBD_4x","Phenotypes_4x")
PVE(IBD_list = IBD_4x,
    Phenotype.df = Phenotypes_4x,
    genotype.ID = "geno",trait.ID = "pheno",
    block = "year",
    QTL_df = data.frame(LG=1,cM=12.3))
```

QTLscan *General QTL function that allows for co-factors, completely randomised block designs and the possibility to derive LOD thresholds using a permutation test*

Description

Function to run QTL analysis using IBD probabilties given (possibly replicated) phenotypes, assuming randomised experimental design

Usage

```
QTLscan(
  IBD_list,
  Phenotype.df,
  genotype.ID,
  trait.ID,
  block = NULL,
  folder = NULL,
  filename.short,
  cofactor_df = NULL,
  prop_Pheno_rep = 0.5,
  perm_test = FALSE,
  N_perm.max = 1000,
  alpha = 0.05,
  gamma = 0.05,
  ncores = 1,
  log = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

IBD_list	List of IBD probabilities
Phenotype.df	A data.frame containing phenotypic values
genotype.ID	The colname of Phenotype.df that contains the offspring identifiers (F1 names)
trait.ID	The colname of Phenotype.df that contains the response variable to use in the model
block	The blocking factor to be used, if any (must be colname of Phenotype.df). By default NULL, in which case no blocking structure (for unreplicated experiments)
folder	If markers are to be used as co-factors, the path to the folder in which the imported IBD probabilities is contained can be provided here. By default this is NULL, if files are in working directory.

filename.short If TetraOrigin was used and co-factors are being included, the shortened stem of
                the filename of the `.csv` files containing the output of TetraOrigin, i.e. without
                the tail "_LinkageGroupX_Summary.csv" which is added by default to all output
                of TetraOrigin.

cofactor_df   A 2-column data frame of co-factor(s); column 1 gives linkage group identifiers,
                column 2 specifies the cM position of the co-factors. By default `NULL`, in which
                case no co-factors are included in the analysis.

prop_Pheno_rep The minimum proportion of phenotypes represented across blocks. If less than
                this, the individual is removed from the analysis. If there is incomplete data,
                the missing phenotypes are imputed using the mean values across the recorded
                observations.

perm_test   Logical, by default `FALSE`. If `TRUE`, a permutation test will be performed to de-
                termine a genome-wide significance threshold.

N_perm.max   The maximum number of permutations to run if `perm_test` is `TRUE`; by default
                this is 1000.

alpha   The P-value to be used in the selection of a threshold if `perm_test` is `TRUE`, by
                default 0.05 (i.e. the 0.95 quantile).

gamma   The width of the confidence intervals used around the permutation test threshold
                using the approach of Nettleton & Doerge (2000), by default 0.05.

ncores   Number of cores to use if parallel computing is required. Works both for Win-
                dows and UNIX (using doParallel). Use `parallel::detectCores()` to find
                out how many cores you have available.

log   Character string specifying the log filename to which standard output should be
                written. If `NULL` log is send to stdout.

verbose   Logical, by default `TRUE`. Should messages be printed during running?

...   Arguments passed to plot

Value

A nested list; each list element (per linkage group) contains the following items:

- QTL.res : Single matrix of QTL results with columns chromosome, position, LOD, adj.r.squared
  and PVE (percentage variance explained).

- Perm.res : If `perm_test = FALSE`, this will be `NULL`. Otherwise, Perm.res contains a list of
  the results of the permutation test, with list items "quantile","threshold" and "scores". Quan-
  tile refers to which quantile of scores was used to determine the threshold. Note that scores
  are each of the maximal LOD scores across the entire genome scan per permutation, thus re-
  turning a genome-wide threshold rather than a chromosome-specific threshold. If the latter is
  preferred, restricting the `IBD_list` to a single chromosome and re-running the permutation
  test will provide the desired threshold.

- Residuals : If a blocking factor or co-factors are used, this is the (named) vector of residuals
  used as input for the QTL scan. Otherwise, this is the set of (raw) phenotypes used in the QTL
  scan.

- Map : Original map of genetic marker positions upon which the IBDs were based, most often
  used for adding rug of marker positions to QTL plots.

- LG_names : Names of the linkage groups

Examples

```
data("IBD_4x","Phenotypes_4x")
qtl_LODs.4x <- QTLscan(IBD_list = IBD_4x,
                        Phenotype.df = Phenotypes_4x,
                        genotype.ID = "geno",
                        trait.ID = "pheno",
                        block = "year")
```

---

qtl_LODs.4x   *A list of QTL results (4x)*

Description

A list of QTL results (4x)

Usage

```
qtl_LODs.4x
```

Format

An object of class list of length 4.

Rec_Data_4x   *A list of recombination count data (4x)*

Description

A list of recombination count data (4x)

Usage

```
Rec_Data_4x
```

Format

An object of class list of length 2.

---

rem.hex   *Redundant genotype classes in hexavalent transition matrix (6x)*

Description

Redundant genotype classes in hexavalent transition matrix (6x)

Usage

```
rem.hex
```

Format

An object of class `integer` of length 166.

---

rem.quad   *Redundant genotype classes in quadrivalent transition matrix (4x)*

Description

Redundant genotype classes in quadrivalent transition matrix (4x)

Usage

```
rem.quad
```

Format

An object of class `integer` of length 6.

---

segList_2x   *A list of all possible bi-allelic QTL segregation types (2x)*

Description

A list of all possible bi-allelic QTL segregation types (2x)

Usage

```
segList_2x
```

Format

An object of class `list` of length 8.

| segList_3x | *A list of all possible bi-allelic QTL segregation types (3x)* |

Description

A list of all possible bi-allelic QTL segregation types (3x)

Usage

```
segList_3x
```

Format

An object of class list of length 27.

| segList_4x | *A list of all possible bi-allelic QTL segregation types (4x)* |

Description

A list of all possible bi-allelic QTL segregation types (4x)

Usage

```
segList_4x
```

Format

An object of class list of length 224.

| segList_6x | *A list of all possible bi-allelic QTL segregation types (6x)* |

Description

A list of all possible bi-allelic QTL segregation types (6x)

Usage

```
segList_6x
```

Format

An object of class list of length 3735.

---

segMaker   *Create a list of possible QTL segregation types*

Description

Function to generate list of segregation types for the [exploreQTL](#) function

Usage

```
segMaker(ploidy, segtypes, modes = c("a", "d"))
```

Arguments

ploidy   The ploidy of the population. Currently assumed to be an even number for this
                function.

segtypes   List of QTL segregation types to consider, so e.g. c(1,0) would mean all pos-
                sible simplex x nulliplex QTL (ie. 4 QTL, on each of homologues 1 - 4 of
                parent 1). Note that symmetrical QTL types that cannot be distinguished are not
                automatically removed and need to be manually identified. If this is an issue,
                use the inbuilt list for tetraploids provided with the package to search the full
                model space. Such an inbuilt list is currently only available for tetraploids, and
                is available from the [exploreQTL](#) function.

modes   Character vector of modes of QTL action to consider, with options "a" for "ad-
                ditive" and "d" for dominant QTL action.

---

singleMarkerRegression

                              Run a single marker regression using marker dosages

---

Description

Function to run a single marker regression using marker dosages

Usage

```
singleMarkerRegression(
  dosage_matrix,
  Phenotype.df,
  genotype.ID,
  trait.ID,
  maplist = NULL,
  perm_test = FALSE,
  N_perm = 1000,
  alpha = 0.05,
  ncores = 1,
```

```
    return_R2 = FALSE,
    log = NULL
)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns. All markers in this matrix will be tested for association with the trait.
Phenotype.df	A data.frame containing phenotypic values
genotype.ID	The colname of Phenotype.df that contains the population identifiers (F1 names) (must be a colname of Phenotype.df)
trait.ID	The colname of Phenotype.df that contains the response variable to use in the model (must be a colname of Phenotype.df)
maplist	Option to include linkage map in the format returned by MDSMap_from_list from polymapR. If maplist is not specified (by default NULL) then no ordering of markers from dosage-matrix is performed. Note that all markers in dosage_matrix are tested; markers with dosages that were not on the maplist will be assigned unordered to linkage group 0 with dummy cM positions 1,2,3 etc.
perm_test	Logical, by default FALSE. If TRUE, a permutation test will be performed to determine a genome-wide significance threshold.
N_perm	Integer. The number of permutations to run if perm_test is TRUE; by default this is 1000.
alpha	Numeric. The P-value to be used in the selection of a threshold if perm_test is TRUE; by default 0.05 (i.e. the 0.95 quantile).
ncores	Number of cores to use if parallel processing required. Works both for Windows and UNIX (using doParallel). Use parallel::detectCores() to find out how many cores you have available.
return_R2	Should the (adjusted) R2 of the model fit also be determined?
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A list containing the following components:

- QTL.res : The -log(p) of the model fit per marker are returned as "LOD" scores, although "LOP" would have been a better description. If requested, R2 values are also returned in column "R2adj"
- Perm.res : The results of the permutation test if performed, otherwise NULL
- Map : The linkage map if provided, otherwise NULL

Examples

```
data("SNP_dosages.4x","BLUEs.pheno")
Trait_1.smr <- singleMarkerRegression(dosage_matrix = SNP_dosages.4x,
Phenotype.df = BLUEs.pheno,genotype.ID = "Geno",trait.ID = "BLUE")
```

SNP_dosages.4x   *A matrix of SNP marker dosages (4x)*

Description

A matrix of SNP marker dosages (4x)

Usage

```
SNP_dosages.4x
```

Format

An object of class matrix (inherits from array) with 186 rows and 52 columns.

---

spline_IBD   *Fit splines to IBD probabilities*

Description

Fits splines to IBD probabilities at a grid of positions at user-defined spacing.

Usage

```
spline_IBD(IBD_list, gap, method = "cubic", ncores = 1, log = NULL)
```

Arguments

IBD_list	List of IBD probabilities
gap	The size (in centiMorgans) of the gap between splined positions
method	One of two options, either "linear" or "cubic". The default method (cubic) fits cubic splines, and although more accurate, becomes computationally expensive in higher-density data-sets, where the linear option may be preferable.
ncores	Number of cores to use, by default 1 only. Works both for Windows and UNIX (using doParallel). Use parallel::detectCores() to find out how many cores you have available. Note that with large datasets, using multiple cores will use large amounts of memory (RAM). Single-core or e.g. 2-core evaluations, although slower, is less memory-intensive.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

Returns a list of similar format as IBD_list, with a splined IBD_array in place of the original IBD_array

Examples

```
data("IBD_4x")
IBD_4x.spl <- spline_IBD(IBD_list = IBD_4x, gap = 1)
```

---

thinmap   *Thin out map data*

Description

`thinmap` is a function for thinning out an integrated map, in order that IBD estimation runs more quickly. Especially useful for maps with very high marker densities for which the estimate_IBD function is to be used.

Usage

```
thinmap(
  maplist,
  dosage_matrix,
  bin_size = 1,
  bounds = NULL,
  remove_markers = NULL,
  plot_maps = TRUE,
  parent1 = "P1",
  parent2 = "P2",
  log = NULL
)
```

Arguments

maplist	A list of maps. In the first column marker names and in the second their position.
dosage_matrix	An integer matrix with markers in rows and individuals in columns.
bin_size	Numeric. Size (in cM) of the bins to include. By default, a bin size of 1 cM is used. Larger `bin_size` results in fewer markers being left on the resulting map.
bounds	Numeric vector. If NULL (by default) then all positions are included, however if specified then output is limited to a specific region, which may be useful if fine-mapping a region of interest.
remove_markers	Optional vector of marker names to remove from the maps. Default is NULL.
plot_maps	Logical. Plot the marker positions of the selected markers using polymapR::plot_map.
parent1	Identifier of parent 1, by default assumed to be "P1"
parent2	Identifier of parent 2, by default assumed to be "P2"
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A maplist of the same structure as the input maplist, but with fewer markers based on the bin_size.

Examples

```
data("phased_maplist.4x","SNP_dosages.4x")
maplist_thin<-thinmap(maplist=phased_maplist.4x,dosage_matrix=SNP_dosages.4x)
```

---

visualiseGIC   *Visualise Genotypic Information Coefficient*

Description

Function to visualise the GIC of a certain region

Usage

```
visualiseGIC(
  GIC_list,
  add_rug = TRUE,
  add_leg = FALSE,
  ylimits = NULL,
  gic.cex = 1,
  show_markers = TRUE,
  add.mainTitle = TRUE,
  plot.cols = NULL
)
```

Arguments

GIC_list	List of GIC data, the output of `estimate_GIC`
add_rug	Should original marker positions be added to the plot?
add_leg	Should a legend be added to the plot?
ylimits	Optional argument to control the plotting area, by default NULL
gic.cex	Option to increase the size of the GIC
show_markers	Should markers be shown?
add.mainTitle	Should a main title be added to the plot?
plot.cols	Optional argument to specify plot colours, otherwise suitable contrasting colours are chosen

Value

The phased map data for the specified region, recoded into 1's and 0's.

Examples

```
data("GIC_4x")
visualiseGIC(GIC_list = GIC_4x)
```

---

visualiseHaplo *Visualise haplotypes in certain individuals in a certain region*

Description

Function to visualise the haplotypes of a certain region in certain individuals

Usage

```
visualiseHaplo(
  IBD_list,
  display_by = c("phenotype", "name"),
  linkage_group = NULL,
  Phenotype.df = NULL,
  genotype.ID = NULL,
  trait.ID = NULL,
  pheno_range = NULL,
  cM_range = "all",
  highlight_region = NULL,
  select_offspring = NULL,
  recombinant_scan = NULL,
  allele_fish = NULL,
  presence_threshold = 0.95,
  xlabl = TRUE,
  ylabl = TRUE,
  mainTitle = NULL,
  multiplot = NULL,
  append = FALSE,
  colPal = c("white", "navyblue", "darkred"),
  hap.wd = 0.4,
  recombination_data = NULL,
  reset_par = TRUE,
  log = NULL
)
```

Arguments

IBD_list   List of IBD probabilities

display_by   Option to display a subset of the population's haplotypes either by "phenotype"
                or "name". If "phenotype" is supplied, then Phenotype.df,genotype.ID,trait.ID
                and pheno_range must also be specified. if "name" is supplied, then select_offspring
                must be specified.

linkage_group	Numeric identifier of the linkage group being examined, based on the order of `IBD_list`. Only a single linkage group is allowed. If `IBD_list` corresponds to a single linkage group, default value of `NULL` will suffice
Phenotype.df	A data.frame containing phenotypic values, which can be used to select a subset of the population to visualise (with extreme phenotypes for example). By default `NULL`, in which case a subset of the population may be selected using the `select_offspring` argument.
genotype.ID	The colname of `Phenotype.df` that contains the population identifiers (F1 names) (must be a colname of `Phenotype.df`)
trait.ID	The colname of `Phenotype.df` that contains the response variable to use in the model (must be a colname of `Phenotype.df`)
pheno_range	Vector of numeric bounds of the phenotypic scores to include (offspring selection).
cM_range	Vector of numeric bounds of the genetic region to be explored. If none are specified, the default of `"all"` means all cM positions will be included.
highlight_region	
	Option to hightlight a particular genetic region on the plot; can be a single position or a vector of 2 positions. By default `NULL`.
select_offspring	
	Vector of offspring identifiers to visualise, must be supplied if `display_by = "name"`. Specifying `"all"` will result in all offspring haplotypes being visualised.
recombinant_scan	
	Vector of homologue numbers between which to search for recombinant offspring in the visualised region and selected individuals. By default `NULL`, in which case no search is preformed.
allele_fish	Vector of homologue numbers of interest, for which to search for offspring that carry these homologues (in the visualised region). By default `NULL`, in which case no search ("fishing") is performed.
presence_threshold	
	Numeric. The minimum probability used to declare presence of a homologue in an individual. This is only needed if a `recombinant_scan` is performed. By default a value of 0.95 is used. When searching for recombinants, this value is also used to denote the proportion of loci carrying the required number of homologues (i.e. by default 95 per cent of loci should have between 0.95 and 1.1 copies of the specified recombinant homologues).
xlabl	Logical, by default `TRUE`. Should an x-axis label be used?
ylabl	Logical, by default `TRUE`. Should a y-axis label be used?
mainTitle	Option to override default plot titles with a (vector of) captions. By default `NULL`.
multiplot	Vector of integers. By default `NULL` so haplotypes are plotted singly; otherwise a vector specifying the number of rows and columns in the plot layout.
append	Option to allow user to append new plots to spaces generated by `multiplot`, otherwise these are filled with blank plots. By default `FALSE`. If `TRUE`, then a large enough `multiplot` grid should be generated to make this option meaningful.

colPal	Colour palette to use in the visualisation (best to provide 3 colours).
hap.wd	The width of the haplotype tracks to be plotted, generally recommended to be about 0.4 (default value)
recombination_data	
	List object as returned by the function `count_recombinations`. By default `NULL`, in which case no overlay of predicted recombination events is performed. However, it can be useful to visualise predicted recombination events, particularly as this might help inform the choice of argument `plausible_pairing_prob` of that function. See [count_recombinations](#) for more details.
reset_par	By default `TRUE`, reset par on exit.
log	Character string specifying the log filename to which standard output should be written. If `NULL` log is send to stdout.

Value

If `recombinant_scan` vector is supplied, a vector of recombinant offspring ID in the region of interest (otherwise `NULL`).

Examples

```
data("IBD_4x")
visualiseHaplo(IBD_list = IBD_4x,
               display_by = "name",
               linkage_group = 1,
               select_offspring = "all",
               multiplot = c(3,3))
```

---

visualisePairing	*Visualise pairing of parental homologues*

Description

Function to visualise the pairing of parental homologues across the population using graph, with nodes to denote parental homologues and edges to denote deviations from expected proportions under a polysomic model of inheritance

Usage

```
visualisePairing(
  meiosis_report.ls,
  pos.col = "red",
  neg.col = "blue",
  parent,
  max.lwd = 20,
  datawidemax,
  add.label = TRUE,
  return.data = FALSE,
  ...
)
```

Arguments

meiosis_report.ls

        List output of function [meiosis_report](#)

pos.col   Colour corresponding to excess of pairing associations predicted (positive deviations), by default red

neg.col   Colour corresponding to lack of pairing associations predicted (negative deviations), by default blue

parent   The parent, either "P1" (mother) or "P2 (father)

max.lwd   Maximum line width, by default 20

datawidemax   This argument is currently a work-around to allow multiple plots to have the same scale (line thicknesses consistent). No default is provided. To estimate this value, simply set argument return.data = TRUE, and record the maximum absolute value over columns 'count', which are the deviations from random expectations. This should be done over multiple function calls if e.g. comparing both P1 and P2 values. When a global maximum (absolute) deviation is known, re-run the function with this value for datawidemax. The line width specified by max.lwd will then be used for this, and all other line widths re-scaled accordingly.

add.label   Should a label be applied, giving the maximum deviation in the plot? By default TRUE

return.data   Should plot data be returned? By default FALSE

...   Optional arguments passed to [plot.igraph](#)

Value

If return.data = TRUE, the values for pairwise deviations from the expected numbers are returned, useful for determining the value datawidemax to provide consistent scaling across multiple plots

Examples

```
data("mr.ls")
visualisePairing(meiosis_report.ls = mr.ls,
                 parent = "P1",
                 datawidemax = 3)
```

---

visualiseQTLeffects   *Visualise QTL homologue effects around a QTL position*

Description

Function to visualise the effect of parental homologues around a QTL peak across the population.

Usage

```
visualiseQTLeffects(
  IBD_list,
  Phenotype.df,
  genotype.ID,
  trait.ID,
  linkage_group,
  LOD_data,
  cM_range = NULL,
  col.pal = c("purple4", "white", "seagreen"),
  point.density = 50,
  zero.sum = FALSE,
  return_plotData = FALSE
)
```

Arguments

`IBD_list`	List of IBD probabilities
`Phenotype.df`	A data.frame containing phenotypic values
`genotype.ID`	The colname of `Phenotype.df` that contains the population identifiers (F1 names) (must be a colname of `Phenotype.df`)
`trait.ID`	The colname of `Phenotype.df` that contains the response variable to use in the model (must be a colname of `Phenotype.df`)
`linkage_group`	Numeric identifier of the linkage group being tested, based on the order of `IBD_list`. Only a single linkage group is allowed.
`LOD_data`	Output of [QTLscan](#) function
`cM_range`	If required, the plotting region can be restricted to a specified range of centiMorgan positions (provided as a vector of start and end positions).
`col.pal`	Vector of colours to use in the visualisations (it is best to provide two or three colours for simplicity). By default, effects will be coloured from purple to green through white.
`point.density`	Parameter to increase the smoothing of homologue effect tracks
`zero.sum`	How allele substitution effect should be defined. If `FALSE` (by default), the effect of each homologue is computed relative to the overall phenotypic mean, otherwise contrasts (against offspring without the inherited homologue) are used.
`return_plotData`	
	Logical, by default `FALSE`. If `TRUE`, plot data is returned, otherwise `NULL`.

Value

The estimated effects of the homologues, used in the visualisation

Examples

```
data("IBD_4x","BLUEs.pheno","qtl_LODs.4x")
visualiseQTLeffects(IBD_list = IBD_4x,
```

```
Phenotype.df = BLUEs.pheno,
genotype.ID = "Geno",
trait.ID = "BLUE",
linkage_group = 2,
LOD_data = qtl_LODs.4x)
```

Index

45