Package 'polymapR'

November 1, 2018

Type Package

Title Linkage Analysis in Outcrossing Polyploids

Version 1.0.18

Date 2018-11-01

Description Creation of linkage maps in polyploid species from marker dosage scores of an F1 cross from two heterozygous parents. Currently works for autotriploid, autotetraploid and autohexaploid species, as well as segmental allotetraploids. Methods are described in a manuscript of Bourke et al. (2018) <doi:10.1093/bioinformatics/bty371>.

Depends R (>= 3.2.0)

License GPL

Imports igraph, doParallel, foreach, knitr, combinat, MDSMap

RoxygenNote 6.0.1

Suggests rmarkdown, Hmisc, LPmerge, reshape2, RColorBrewer

VignetteBuilder knitr

NeedsCompilation no

Author Peter Bourke [aut, cre], Geert van Geest [aut]

Maintainer Peter Bourke <pbourkey@gmail.com>

Repository CRAN

Date/Publication 2018-11-01 22:20:03 UTC

R topics documented:

`add_dup_markers` *Add back duplicate markers after mapping*

Description

Often there will be duplicate markers that can be put aside to speed up mapping. These may be added back to the maps afterwards.

Usage

```
add_dup_markers(maplist, bin_list, marker_assignments = NULL)
```

Arguments

`maplist`	A list of maps. Output of MDSMap_from_list.
`bin_list`	A list of marker bins containing marker duplicates. One of the list outputs of `screen_for_duplicate_markers`
`marker_assignments`	Optional argument to include the marker_assignments (output of `check_marker_assignment`). If included, marker assignment information will also be copied.

Value

A list with the following items:

- maplist: List of maps, now with duplicate markers added
- marker_assignments: If required, marker assignment list with duplicate markers added

ALL_dosages — *A dosage matrix for a random pairing tetraploid with five linkage groups.*

Description

A dosage matrix for a random pairing tetraploid with five linkage groups.

Usage

```
ALL_dosages

segregating_data

screened_data

screened_data2

screened_data3

TRI_dosages
```

Format

A matrix

all_linkages_list_P1 — *A (nested) list of linkage data frames classified per linkage group and homologue*

Description

A (nested) list of linkage data frames classified per linkage group and homologue

Usage

```
all_linkages_list_P1

all_linkages_list_P1_split

all_linkages_list_P1_subset
```

Format

An object of class `list` of length 5.

assign_linkage_group *Assign non-SN markers to a linkage group and homologue(s).*

Description

assign_linkage_group quantifies per marker number of linkages to a linkage group and evaluates to which linkage group (and homologue(s)) the marker belongs.

Usage

```
assign_linkage_group(linkage_df, LG_hom_stack, SN_colname = "marker_a",
  unassigned_marker_name = "marker_b", phase_considered = "coupling",
  LG_number = 12, LOD_threshold = 3, ploidy = 4, assign_homologue = T,
  log = NULL)
```

Arguments

linkage_df	A linkage data.frame as output of linkage.
LG_hom_stack	A data.frame with markernames ("SxN_Marker"), linkage group ("LG") and homologue ("homologue")
SN_colname	The name of the column in linkage_df harbouring the 1.0 markers
unassigned_marker_name	The name of the column in linkage_df harbouring the marker that are to be assigned.
phase_considered	The phase that is used to assign the markers (deprecated)
LG_number	The number of chromosomes (linkage groups) in the species.
LOD_threshold	The LOD score at which a linkage to a linkage group is significant.
ploidy	The ploidy of the plant species.
assign_homologue	Logical. Should markers be assigned to homologues? If FALSE markers will be assigned to all homologues
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

Output is a data.frame with at least the following columns:

Assigned_LG	The assigned linkage group
Assigend_hom1	The homologue with most linkages

The columns LG1 - LGn and Hom1 - Homn give the number of hits per marker for that linkage group/homologue. Assigned_hom2 .. gives the nth homologue with most linkages.

Examples

```
data("SN_DN_P1", "LGHomDf_P1_1")
assigned_df<-assign_linkage_group(linkage_df = SN_DN_P1,
                     LG_hom_stack = LGHomDf_P1_1,
                     LG_number = 5)
```

`assign_SN_SN` *Assign (leftover) 1.0 markers*

Description

Some 1.0 markers might have had ambiguous linkages, or linkages with low LOD scores leaving them unlinked to a linkage group. `assign_SN_SN` finds 1.0 markers unlinked to a linkage group and tries to assign them.

Usage

```
assign_SN_SN(linkage_df, LG_hom_stack, LOD_threshold, ploidy, LG_number,
  log = NULL)
```

Arguments

`linkage_df`	A `data.frame` as output of `linkage` with arguments markertype1=c(1,0) and markertype2=NULL.
`LG_hom_stack`	A `data.frame` with markernames (`"SxN_Marker"`), linkage group (`"LG"`) and homologue (`"homologue"`)
`LOD_threshold`	A LOD score at which linkages between markers are significant.
`ploidy`	Integer. The ploidy level of the plant species.
`LG_number`	Integer. Number of chromosomes (linkage groups)
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

Returns a `data.frame` with the following columns:

`SxN_Marker`	The markername
`Assigned_hom1`	The assigned homologue
`Assigned_LG`	The assigned linkage group

Examples

```
data("SN_SN_P1", "LGHomDf_P1_1")
SN_assigned<-assign_SN_SN(linkage_df = SN_SN_P1,
             LG_hom_stack = LGHomDf_P1_1,
             LOD_threshold= 4,
             ploidy=4,
             LG_number=5)
```

`bridgeHomologues` *Use bridge markers to cluster homologues into linkage groups*

Description

Clustering at high LOD scores results in marker clusters representing homologues. `bridgeHomologues` clusters these (pseudo)homologues to linkage groups using linkage information between 1.0 and bridge markers within a parent (e.g. 2.0 for a tetraploid). If parent-specific bridge markers (e.g. 2.0) cannot be used, biparental markers can also be used (e.g. 1.1, 1.2, 2.1, 2.2 and 1.3 markers). The linkage information between 1.0 and biparental markers can be combined.

Usage

```
bridgeHomologues(cluster_stack, cluster_stack2 = NULL, linkage_df,
  linkage_df2 = NULL, LOD_threshold = 5, automatic_clustering = TRUE,
  LG_number = 5, parentname = "", min_links = 1, min_bridges = 1,
  only_coupling = FALSE, log = NULL)
```

Arguments

`cluster_stack`	A `data.frame` with a column `"marker"` specifying markernames, and a column `"cluster"` specifying marker cluster
`cluster_stack2`	Optional. A `cluster_stack` for the other parent. Use this argument if cross-parent markers are used (e.g. when using 1.1 markers).
`linkage_df`	A linkage `data.frame` as output of `linkage` between bridge (e.g. 1.0 and 2.0) markers.
`linkage_df2`	Optional. A `linkage_df` specifying linkages between 1.0 and cross-parent markers in the other parent. Use this argument if cross-parent markers are used (e.g. when using 1.1, 2.1, 1.2 and/or 2.2 markers). The use of multiple types of cross-parent markers is allowed.
`LOD_threshold`	Integer. The LOD threshold specifying at which LOD score a link between 1.0 and bridge (e.g. 2.0) markers is used for clustering homologues.
`automatic_clustering`	Logical. Should clustering be executed without user input?
`LG_number`	Integer. Expected number of chromosomes (linkage groups)
`parentname`	Name of the parent. Used in the main title of the plot.
`min_links`	The minimum number of cross-parent linkages for a marker to be considered. Make this number higher if there are a lot of spurious links.
`min_bridges`	The minimum number of linking markers to link two homologues together.
`only_coupling`	Logical, should only coupling linkages be used in the process? By default `FALSE`
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A data.frame with markers classified by homologue and linkage group.

Examples

```
data("P1_homologues", "P2_homologues", "SN_DN_P1", "SN_SS_P1", "SN_SS_P2")
ChHomDf<-bridgeHomologues(cluster_stack = P1_homologues[["5"]],
                 linkage_df=SN_DN_P1,
                 LOD_threshold=4,
                 automatic_clustering=TRUE,
                 LG_number=5,
                 parentname="P1")

ChHomDf<-bridgeHomologues(cluster_stack = P1_homologues[["5"]],
                           cluster_stack2 = P2_homologues[["5"]],
                 linkage_df=SN_SS_P1,
                 linkage_df2=SN_SS_P2,
                 LOD_threshold=4,
                 automatic_clustering=TRUE,
                 LG_number=5,
                 parentname="P1")
```

---

calcSegtypeInfo *Build a list of segregation types*

Description

For each possible segregation type in an F1 progeny with given parental ploidy (and ploidy2, if parent2 has a different ploidy than parent1) information is given on the segregation ratios, parental dosages and whether the segregation is expected under polysomic, disomic and/or mixed inheritance.

Usage

```
calcSegtypeInfo(ploidy, ploidy2=NULL)
```

Arguments

ploidy	The ploidy of parent 1 (must be even, 2 (diploid) or larger).
ploidy2	The ploidy of parent 2. If omitted (default=NULL) it is assumed to be equal to ploidy.

Details

The names of the segregation types consist of a short sequence of digits (and sometimes letters), an underscore and a final number. This is interpreted as follows, for example segtype 121_0: 121 means that there are three consecutive dosages in the F1 population with frequency ratios 1:2:1, and the 0 after the underscore means that the lowest of these dosages is nulliplex. So 121_0 means

a segregation of 1 nulliplex : 2 simplex : 1 duplex. A monomorphic F1 (one single dosage) is indicated as e.g. 1_4 (only one dosage, the 4 after the underscore means that this is monomorphic quadruplex). If UPPERCASE letters occur in the first part of the name these are interpreted as additional digits with values of A=10 to Z=35, e.g. 18I81_0 means a segregation of 1:8:18:8:1 (using the I as 18), with the lowest dosage being nulliplex.
With higher ploidy levels higher numbers (above 35) may be required. In that case each unique ratio number above 35 is assigned a lowercase letter. E.g. one segregation type in octaploids is 9bcb9_2: a 9:48:82:48:9 segregation where the lowest dosage is duplex.
Segregation types with more than 5 dosage classes are considered "complex" and get codes like c7e_1 (again in octoploids): this means a complex type (the first c) with 7 dosage classes; the e means that this is the fifth type with 7 classes. Again the _1 means that the lowest dosage is simplex. It is always possible (and for all segtype names with lowercase letters it is necessary) to look up the actual segregation ratios in the intratio item of the segtype. For octoploid segtype c7e_1 this shows 0:1:18:69:104:69:18:1:0 (the two 0's mean that nulli- and octoplexes do not occur).

Value

A list with for each different segregation type (segtype) one item. The names of the items are the names of the segtypes. Each item is itself a list with components:

- freq: a vector of the ploidy+1 fractions of the dosages in the F1
- intratios: an integer vector with the ratios as the simplest integers
- expgeno: a vector with the dosages present in this segtype
- allfrq: the allele frequency of the dosage allele in the F1
- polysomic: boolean: does this segtype occur with polysomic inheritance?
- disomic: boolean: does this segtype occur with disomic inheritance?
- mixed: boolean: does this segtype occur with mixed inheritance (i.e. with polysomic inheritance in one parent and disomic inheritance in the other)?
- pardosage: integer matrix with 2 columns and as many rows as there are parental dosage combinations for this segtype; each row has one possible combination of dosages for parent 1 (1st column) and parent 2 (2nd column)
- parmode: logical matrix with 3 columns and the same number of rows as pardosage. The 3 columns are named polysomic, disomic and mixed and tell if this parental dosage combination will generate this segtype under polysomic, disomic and mixed inheritance

Examples

```
si4 <- calcSegtypeInfo(ploidy=4) # two 4x parents: a 4x F1 progeny
print(si4[["11_0"]])

si3 <- calcSegtypeInfo(ploidy=4, ploidy2=2) # a 4x and a diplo parent: a 3x progeny
print(si3[["11_0"]])
```

checkF1 *Identify the best-fitting F1 segregation types*

Description

For a given set of F1 and parental samples, this function finds the best-fitting segregation type. It can perform a dosage shift prior to selecting the segregation type.

Usage

```
checkF1(dosage_matrix, parent1, parent2, F1, ancestors=character(0),
polysomic, disomic, mixed, ploidy, ploidy2, outfile,
critweight=c(1.0, 0.4, 0.4),
Pvalue_threshold=0.0001, fracInvalid_threshold=0.05,
fracNA_threshold=0.25, shiftmarkers, parentsScoredWithF1=TRUE,
shiftParents=parentsScoredWithF1, showAll=FALSE, append_shf=FALSE)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns.
parent1	character vector with the sample names of parent 1
parent2	character vector with the sample names of parent 2
F1	character vector with the sample names of the F1 individuals
ancestors	character vector with the sample names of any other ancestors or other samples of interest. The dosages of these samples will be shown in the output (shifted if shiftParents TRUE) but they are not used in the selection of the segregation type.
polysomic	if TRUE at least all polysomic segtypes are considered; if FALSE these are not specifically selected (but if e.g. disomic is TRUE, any polysomic segtypes that are also disomic will still be considered)
disomic	if TRUE at least all disomic segtypes are considered (see polysomic)
mixed	if TRUE at least all mixed segtypes are considered (see polysomic). A mixed segtype occurs when inheritance in one parent is polysomic (random chromosome pairing) and in the other parent disomic (fully preferential chromosome pairing)
ploidy	The ploidy of parent 1 (must be even, 2 (diploid) or larger).
ploidy2	The ploidy of parent 2. If omitted it is assumed to be equal to ploidy.
outfile	the file to write the output to; if NA a temporary file checkF1.tmp is created in the current working directory and deleted at end
critweight	NA or a numeric vector containing the weights of three quality criteria; do not need to sum to 1. If NA, the output will not contain a column qall_weights. Else the weights specify how qall_weights will be calculated from quality parameters q1, q2 and q3.

`Pvalue_threshold`
: a minimum threshold value for the Pvalue of the bestParentfit segtype (with a smaller Pvalue the q1 quality parameter will be set to 0)

`fracInvalid_threshold`
: a maximum threshold for the fracInvalid of the bestParentfit segtype (with a larger fraction of invalid dosages in the F1 the q1 quality parameter will be set to 0)

`fracNA_threshold`
: a maximum threshold for the fraction of unscored F1 samples (with a larger fraction of unscored samples in the F1 the q3 quality parameter will be set to 0)

`shiftmarkers`
: if specified, shiftmarkers must be a data frame with columns MarkerName and shift; for the markernames that match exactly (upper/lowercase etc) those in `dosage_matrix`, the dosages are increased by the amount specified in column shift, e.g. if shift is -1, dosages 2..ploidy are converted to 1..(ploidy-1) and dosage 0 is a combination of old dosages 0 and 1, for all samples. The segregation check is then performed with the shifted dosages. A shift=NA is allowed, these markers will not be shifted. The sets of markers in `dosage_matrix` and shiftmarkers may be different, but markers may occur only once in shiftmarkers. A column shift is added at the end of the returned data frame.
If parameter shiftParents is `TRUE`, the parental and ancestor scores are shifted as the F1 scores, if `FALSE` they are not shifted.

`parentsScoredWithF1`
: `TRUE` if parents are scored in the same experiment and the same `fitPoly` run as the F1, else `FALSE`. If `TRUE`, their fraction missing scores and conflicts tell something about the quality of the scoring. If `FALSE` (e.g. when the F1 is triploid and the parents are diploid and tetraploid) the quality of the F1 scores can be independent of that of the parents

`shiftParents`
: only used if parameter shiftmarkers is specified. If `TRUE`, apply the shifts also to the parental and ancestor scores. By default `TRUE` if `parentsScoredWithF1` is `TRUE`

`showAll`
: (default `FALSE`) if `TRUE`, for each segtype 3 columns are added to the returned data frame with the frqInvalid, Pvalue and matchParents values for these segtype (see the description of the return value)

`append_shf`
: if `TRUE` and parameter shiftmarkers is specified, _shf is appended to all marker names where shift is not 0. This is not required for any of the functions in this package but may prevent duplicated marker names when using other software.

Details

For each marker is tested how well the different segregation types fit with the observed parental and F1 dosages. The results are summarized by columns bestParentfit (which is the best fitting segregation type, taking into account the F1 and parental dosages) and columns qall_mult and/or qall_weights (how good is the fit of the bestParentfit segtype: 0=bad, 1=good).
Column bestfit in the results gives the segtype best fitting the F1 segregation without taking account of the parents. This bestfit segtype is used by function correctDosages, which tests for possible "shifts" in the marker models.
In case the parents are not scored together with the F1 (e.g. if the F1 is triploid and the parents are diploid and tetraploid) `dosage_matrix` should be edited to contain the parental as well

as the F1 scores. In case the diploid and tetraploid parent are scored in the same run of function `saveMarkerModels` (from package `fitPoly`) the diploid is initially scored as nulliplex-duplex-quadruplex (dosage 0, 2 or 4); that must be converted to the true diploid dosage scores (0, 1 or 2). Similar corrections are needed with other combinations, such as a diploid parent scored together with a hexaploid population etc.

Value

A data frame with one row per markers, with the following columns:

- m: the sequential number of the marker (as assigned by `fitPoly`)
- MarkerName: the name of the marker, with _shf appended if the marker is shifted and append_shf is `TRUE`
- parent1: consensus dosage score of the samples of parent 1
- parent2: consensus dosage score of the samples of parent 2
- F1_0 ... F1_<ploidy>: the number of F1 samples with dosage scores 0 ... <ploidy>
- F1_NA: the number of F1 samples with a missing dosage score
- sample names of parents and ancestors: the dosage scores for those samples
- bestfit: the best fitting segtype, considering only the F1 samples
- frqInvalid_bestfit: for the bestfit segtype, the frequency of F1 samples with a dosage score that is invalid (that should not occur). The frequency is calculated as the number of invalid samples divided by the number of non-NA samples
- Pvalue_bestfit: the chisquare test P-value for the observed distribution of dosage scores vs the expected fractions. For segtypes where only one dosage is expected (1_0, 1_1 etc) the binomial probability of the number of invalid scores is given, assuming an error rate of seg_invalidrate (hard-coded as 0.03)
- matchParent_bestfit: indication how the bestfit segtype matches the consensus dosages of parent 1 and 2: "Unknown"=both parental dosages unknown; "No"=one or both parental dosages known and conflicting with the segtype; "OneOK"= only one parental dosage known, not conflicting with the segtype; "Yes"=both parental dosages known and combination matching with the segtype. This score is initially assigned based on only high-confidence parental consensus scores; if low-confidence dosages are confirmed by the F1, the matchParent for (only) the selected segtype is updated, as are the parental consensus scores.
- bestParentfit: the best fitting segtype that does not conflict with the parental consensus scores
- frqInvalid_bestParentfit, Pvalue_bestParentfit, matchParent_bestParentfit: same as the corresponding columns for bestfit. Note that matchParent_bestParentfit cannot be "No".
- q1_segtypefit: a value from 0 (bad) to 1 (good), a measure of the fit of the bestParentfit segtype based on Pvalue, invalidP and whether bestfit is equal to bestParentfit
- q2_parents: a value from 0 (bad) to 1 (good), based either on the quality of the parental scores (the number of missing scores and of conflicting scores, if parentsScoredWithF1 is TRUE) or on matchParents (No=0, Unknown=0.65, OneOK=0.9, Yes=1, if parentsScoredWithF1 is FALSE)
- q3_fracscored: a value from 0 (bad) to 1 (good), based on the fraction of F1 samples that have a non-missing dosage score

- qall_mult: a value from 0 (bad) to 1 (good), a summary quality score equal to the product q1*q2*q3. Equal to 0 if any of these is 0, hence sensitive to thresholds; a natural selection criterion would be to accept all markers with qall_mult > 0
- qall_weights: a value from 0 (bad) to 1 (good), a weighted average of q1, q2 and q3, with weights as specified in parameter critweight. This column is present only if critweight is specified. In this case there is no "natural" threshold; a threshold for selection of markers must be obtained by inspecting XY-plots of markers over a range of qall_weights values
- shift: if shiftmarkers is specified a column shift is added with for all markers the applied shift (for the unshifted markers the shift value is 0)

qall_mult and/or qall_weights can be used to compare the quality of the SNPs within one analysis and one F1 population but not between analyses or between different F1 populations.
If parameter showAll is TRUE there are 3 additional columns for each segtype with names frqInvalid_<segtype>, Pvalue_<segtype> and matchParent_<segtype>; see the corresponding columns for bestfit for an explanation. These extra columns are inserted directly before the bestfit column.

`check_map` *Check the quality of a linkage map using heatplots*

Description

Perform a series of checks on a linkage map and visualise the results using heatplots. Also shows the discrepency between the pairwise and multi-point r estimates, plotted against the LOD of the pairwise estimate.

Usage

```
check_map(linkage_list, maplist, mapfn = "haldane", lod.thresh = 5)
```

Arguments

`linkage_list`	A named `list` with r and LOD of markers within linkage groups.
`maplist`	A list of maps. In the first column marker names and in the second their position.
`mapfn`	The map function used in generating the maps, either one of "haldane" or "kosambi". By default "haldane" is assumed.
`lod.thresh`	Numeric. Threshold for the LOD values to be displayed in heatmap, by default 5 (set at 0 to display all values)

Examples

```
## Not run:
data("maplist_P1","all_linkages_list_P1")
check_map(linkage_list = all_linkages_list_P1, maplist = maplist_P1)

## End(Not run)
```

check_marker_assignment *Check for consistent marker assignment between both parents*

Description

Function to ensure there is consistent marker assignment to chromosomal linkage groups for bi-parental markers

Usage

```
check_marker_assignment(marker_assignment.P1, marker_assignment.P2,
  log = NULL, verbose = TRUE)
```

Arguments

marker_assignment.P1	A marker assignment matrix for parent 1 with markernames as rownames and at least containing the column "Assigned_LG"; the output of homologue_lg_assignment.
marker_assignment.P2	A marker assignment matrix for parent 2 with markernames as rownames and at least containing the column "Assigned_LG"; the output of homologue_lg_assignment.
log	Character string specifying the log filename to which standard output should be written. If NULL (by default) log is send to stdout.
verbose	Should messages be send to stdout or log?

Value

Returns a list of matrices with corrected marker assignments.

Examples

```
data("marker_assignments_P1"); data("marker_assignments_P2")
check_marker_assignment(marker_assignments_P1,marker_assignments_P2)
```

cluster_per_LG *Cluster 1.0 markers into correct homologues per linkage group*

Description

Clustering at one LOD score for all markers does usually not result in correct classification of homologues. Usually there are more clusters of (pseudo)homologues than expected. This function lets you inspect every linkage group separately and allows for clustering at a different LOD threshold per LG.

Usage

```
cluster_per_LG(LG, linkage_df, LG_hom_stack, LOD_sequence,
  modify_LG_hom_stack = FALSE, nclust_out = NULL,
  network.layout = c("circular", "stacked", "n"), device = NULL,
  label.offset = 1, cex.lab = 0.7, log = NULL, ...)
```

Arguments

LG	Integer. Linkage group to investigate.
linkage_df	A data.frame as output of linkage with arguments markertype1 = c(1,0) and markertype2=NULL.
LG_hom_stack	A data.frame with columns "SxN_Marker" providing 1.0 markernames and "LG" and "homologue" providing linkage group and homologue respectively.
LOD_sequence	A numeric or vector of numerics giving LOD threshold(s) at which clustering should be performed.
modify_LG_hom_stack	Logical. Should LG_hom_stack be modified and returned?
nclust_out	Number of clusters in the output. If there are more clusters than this number only the nclust_out largest clusters are returned.
network.layout	Network layout: "circular" or "stacked". If "n" no network is plotted.
device	Function of the graphics device to plot to (e.g. pdf, png, jpeg). The active device is used when NULL
label.offset	Offset of labels. Only used if network.layout="circular".
cex.lab	label character expansion. Only for network.layout="circular".
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
...	Arguments passed to device.

Value

A modified LG_hom_stack data.frame if modify_LG_hom_stack = TRUE

Examples

```
data("SN_SN_P2", "LGHomDf_P2_1")
#take only markers in coupling:
SN_SN_P2_coupl <- SN_SN_P2[SN_SN_P2$phase=="coupling",]
cluster_per_LG(LG = 2,
               linkage_df=SN_SN_P2_coupl,
               LG_hom_stack=LGHomDf_P2_1,
               LOD_sequence=seq(4,10,2),
               modify_LG_hom_stack=FALSE,
               nclust_out=4,
               network.layout="circular",
               device=NULL,
               label.offset=1.2,
               cex.lab=0.75)
```

`cluster_SN_markers` *Cluster 1.0 markers*

Description

`cluster_SN_markers` clusters simplex nulliplex at different LOD scores.

Usage

```
cluster_SN_markers(linkage_df, LOD_sequence = 7, independence_LOD = FALSE,
  LG_number = 5, ploidy = 4, parentname = "", plot_network = F,
  min_clust_size = 1, plot_clust_size = TRUE, max_vertex_size = 5,
  min_vertex_size = 2, phase_considered = "All", log = NULL)
```

Arguments

`linkage_df`	A linkage data.frame as output of `linkage` calculating linkage between 1.0 markers.
`LOD_sequence`	A numeric vector. Specifying a sequence of LOD thresholds at which clustering is performed.
`independence_LOD`	Logical. Should the LOD of independence be used for clustering? (by default, `FALSE`.)
`LG_number`	Expected number of chromosomes (linkage groups)
`ploidy`	Ploidy level of the plant species
`parentname`	Name of parent
`plot_network`	Logical. Should a network be plotted. Recommended FALSE with large number of marker combinations.
`min_clust_size`	Integer. The minimum cluster size to be plotted. This does not delete clusters. All clusters are returned.
`plot_clust_size`	Logical. Should exact cluster size be plotted as vertex labels?
`max_vertex_size`	Integer. The maximum vertex size. Only used if `plot_clust_size=FALSE`.
`min_vertex_size`	Integer. The minimum vertex size. Only used if `plot_clust_size=FALSE`.
`phase_considered`	Character string. By default all phases are used, but "coupling" or "repulsion" are also allowed.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout (console).

Value

A list with cluster data.frames.

Examples

```
data("SN_SN_P1")
cluster_list<-cluster_SN_markers(SN_SN_P1, LOD_sequence=c(4:10), parentname="P1")
```

`consensus_LG_assignment` *Consensus LG assignment*

Description

Assign markers to an LG based on consensus between two parents.

Usage

```
consensus_LG_assignment(P1_assigned, P2_assigned, LG_number = 5, ploidy = 4,
  consensus_file = NULL, log = NULL)
```

Arguments

`P1_assigned`	A marker assignment file of the first parent. Should contain the number of linkages per LG per marker.
`P2_assigned`	A marker assignment file of the second parent. Should be the same markertype as first parent and contain the number of linkages per LG per marker.
`LG_number`	Number of linkage groups (chromosomes).
`ploidy`	Ploidy level of plant species.
`consensus_file`	Filename of consensus output. No output is written if NULL.
`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 containing the following components:

`P1_assigned`	A (modified) marker assignment matrix of the first parent.
`P2_assigned`	A (modified) marker assignment matrix of the second parent.

Examples

```
data("P1_SxS_Assigned", "P2_SxS_Assigned_2")
SxS_Assigned_list <- consensus_LG_assignment(P1_SxS_Assigned, P2_SxS_Assigned_2, LG_number=5)
```

`consensus_LG_names` *Find consensus linkage group names*

Description

Chromosomes that should have same number, might have gotten different numbers between parents during clustering. `consensus_LG_names` uses markers present in both parents (usually 1.1 markers) to modify the linkage group numbers in one parent with the other as template

Usage

```
consensus_LG_names(modify_LG, template_SxS, modify_SxS, merge_LGs = TRUE,
  log = NULL)
```

Arguments

`modify_LG`	A `data.frame` with markernames, linkage group (`"LG"`) and homologue (`"homologue"`), in which the linkage group numbers will be modified
`template_SxS`	A file with assigned markers of which (at least) part is present in both parents of the template parent.
`modify_SxS`	A file with assigned markers of which (at least) part is present in both parents of the parent of which linkage group number are modified.
`merge_LGs`	Logical, by default `TRUE`. If `FALSE`, any discrepency in the number of linkage groups will not be merged, but removed instead. This can be needed if the number of chromosomes identified is not equal between parents, and the user wishes to proceed with a core set.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A modified modified_LG according to the template_SxS linkage group numbering

Examples

```
data("LGHomDf_P2_2", "P1_SxS_Assigned", "P2_SxS_Assigned")
consensus_LGHomDf<-consensus_LG_names(LGHomDf_P2_2, P1_SxS_Assigned, P2_SxS_Assigned)
```

convert_marker_dosages *Convert marker dosages to the basic types.*

Description

Convert marker dosages to the basic types which hold the same information and for which linkage calculations can be performed.

Usage

```
convert_marker_dosages(dosage_matrix, outname, ploidy = 4, ploidy2 = NULL,
  parent1 = "P1", parent2 = "P2", marker_conversion_info = FALSE,
  log = NULL)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns.
outname	output filename (deprecated for now)
ploidy	ploidy level of the plant species. If parents have different ploidy level, ploidy of parent1.
ploidy2	ploidy level of the second parent. NULL if both parents have the same ploidy level.
parent1	Character string specifying the first (usually maternal) parentname.
parent2	Character string specifying the second (usually paternal) parentname.
marker_conversion_info	Logical. Should marker conversion information be returned?
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A modified dosage matrix. If `marker_swap_info = TRUE`, this function returns a list.

Examples

```
data("ALL_dosages")
conv<-convert_marker_dosages(dosage_matrix=ALL_dosages)
```

`correctDosages` *Check if dosage scores may have to be shifted*

Description

fitPoly sometimes uses a "shifted" model to assign dosage scores (e.g. all samples are assigned a dosage one higher than the true dosage). This happens mostly when there are only few dosages present among the samples. This function checks if a shift of +/-1 is possible.

Usage

```
correctDosages(chk, dosage_matrix, parent1, parent2, ploidy,
polysomic=TRUE, disomic=FALSE, mixed=FALSE,
absent.threshold=0.04)
```

Arguments

`chk`	data frame returned by function checkF1 when called without shiftmarkers
`dosage_matrix`	An integer matrix with markers in rows and individuals in columns.
`parent1`	character vector with names of the samples of parent 1
`parent2`	character vector with names of the samples of parent 2
`ploidy`	ploidy of parents and F1 (correctDosages must not be used for F1 populations where the parents have a different ploidy, or where the parental genotypes are not scored together with the F1); same as used in the call to checkF1 that generated data.frame chk
`polysomic`	if TRUE at least all polysomic segtypes are considered; if FALSE these are not specifically selected (but if e.g. disomic is TRUE, any polysomic segtypes that are also disomic will still be considered); same as used in the call to checkF1 that generated data.frame chk
`disomic`	if TRUE at least all disomic segtypes are considered (see param polysomic); same as used in the call to checkF1 that generated data.frame chk
`mixed`	if TRUE at least all mixed segtypes are considered (see param polysomic). A mixed segtype occurs when inheritance in one parent is polysomic (random chromosome pairing) and in the other parent disomic (fully preferential chromosome pairing); same as used in the call to checkF1 that generated data.frame chk
`absent.threshold`	the threshold for the fraction of ALL samples that has the dosage that is assumed to be absent due to mis-fitting of fitPoly; should be at least the assumed error rate of the fitPoly scoring assuming the fitted model is correct

Details

A shift of -1 (or +1) is proposed when (1) the fraction of all samples with dosage 0 (or ploidy) is below absent.threshold, (2) the bestfit (not bestParentfit!) segtype in chk has one empty dosage on the low (or high) side and more than one empty dosage at the high (or low) side, and (3) the shifted consensus parental dosages do not conflict with the shifted segregation type.
The returned data.frame (or a subset, e.g. based on the values in the fracNotOk and parNA columns) can serve as parameter shiftmarkers in a new call to checkF1.
Based on the quality scores assigned by checkF1 to the original and shifted versions of each marker the user can decide if either or both should be kept. A data.frame combining selected rows of the original and shifted versions of the checkF1 output (which may contain both a shifted and an unshifted version of some markers) can then be used as input to compareProbes or writeDosagefile.

Value

a data frame with columns

- markername
- segtype: the bestfit (not bestParentfit!) segtype from chk
- parent1, parent2: the consensus parental dosages; possibly low-confidence, so may be different from those reported in chk
- shift: -1, 0 or 1: the amount by which this marker should be shifted

The next fields are only calculated if shift is not 0:

- fracNotOk: the fraction of ALL samples that are in the dosage (0 or ploidy) that should be empty if the marker is indeed shifted.
- parNA: the number of parental dosages that is missing (0, 1 or 2)

`createMap` *Marker ordering function*

Description

Creates a linkage map from a .pwd file using the weighted regression algorithm employed by JoinMap

Usage

```
createMap(pwdDATA, parent_ID = "P1", chm_num, h_num, mapFun = c("haldane",
  "kosambi"), jumpThresh = 5, max_rf = 0.4, min_LOD = 1, rippleFREQ = 1,
  rippleRounds = 3, round3 = TRUE, printMAPS = FALSE, log = NULL)
```

Arguments

pwdDATA	pwd data.frame giving pairwise r and LOD estimates.
parent_ID	Identifier of the parent for which the map belongs
chm_num	The number of the chromosome being mapped
h_num	The number of the homologue being mapped
mapFun	The mapping function to use. Currently Haldane and Kosambi are available.
jumpThresh	The "Jump" threshold to use (normalised comparison of G2 values), with default value 5 as employed by JoinMap.
max_rf	The maximum recombination frequency to use in the mapping, default 0.4
min_LOD	The minimum LOD to use in the mapping, default is 1
rippleFREQ	How often to ripple (every added marker? every 2 markers?)
rippleRounds	The number of rounds of rippling to be attempted
round3	Option to stop mapping after two rounds. Default is TRUE, so 3 rounds.
printMAPS	Allows the user to see maps developing, default is FALSE.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Examples

```
## Not run:
data("all_linkages_list_P1_subset")
map_P1_LG2_hom3 <- createMap(pwdDATA=all_linkages_list_P1_subset[["LG2"]][["homologue3"]],
                             parent_ID = "P1",
                             chm_num=2,
                             h_num=3,
                             mapFun = "haldane",
                             jumpThresh = 5,
                             max_rf = 0.4,
                             min_LOD = 1,
                             rippleFREQ = 1,
                             rippleRounds = 3,
                             round3 = TRUE,
                             printMAPS = FALSE,
                             log = NULL)

## End(Not run)
```

---

createTetraOriginInput

Create input files for TetraOrigin using an integrated linkage map list and marker dosage matrix

---

Description

createTetraOriginInput is a function for creating an input file for TetraOrigin, combining map positions with marker dosages.

Usage

```
createTetraOriginInput(maplist, dosage_matrix, bin_size = NULL,
  bounds = NULL, remove_markers = NULL, outdir = "TetraOrigin",
  output_stem = "TetraOrigin_input", plot_maps = TRUE, 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. Either provide the unconverted dosages (i.e. before using the convert_marker_dosages function), or converted dosages (i.e. screened data), in matrix form. The analysis and results are unaffected by this choice, but it may be simpler to understand the results if converted dosages are used. Conversely, it may be advantageous to use the original unconverted dosages if particular marker alleles are being tracked for (e.g.) the development of selectable markers afterwards.
bin_size	Numeric. Size (in cM) of the bins to include. If NULL (by default) then all markers are used (no binning).
bounds	Numeric vector. If NULL (by default) then all positions are included, however if specified then output is limited to a specific region, which is useful for later fine-mapping work.
remove_markers	Optional vector of marker names to remove from the maps. Default is NULL.
outdir	Output directory to which input files for TetraOrigin are written.
output_stem	Character prefix to add to the .csv output filename.
plot_maps	Logical. Plot the marker positions of the selected markers using plot_map.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Examples

```
data("integrated.maplist","ALL_dosages")
createTetraOriginInput(maplist=integrated.maplist,dosage_matrix=ALL_dosages,bin_size=10)
```

---

create_phased_maplist *Create a phased homologue map list using the original dosages*

Description

create_phased_maplist is a function for creating a phased maplist, using integrated map positions and original marker dosages.

Usage

```
create_phased_maplist(maplist, dosage_matrix.conv, dosage_matrix.orig = NULL,
  remove_markers = NULL, N_linkages = 2, lower_bound = 0.05, ploidy = 4,
  ploidy2 = NULL, marker_assignment.1, marker_assignment.2, parent1 = "P1",
  parent2 = "P2", original_coding = FALSE, log = NULL, verbose = TRUE)
```

Arguments

maplist	A list of maps. In the first column marker names and in the second their position.
dosage_matrix.conv	Matrix of marker dosage scores with markers in rows and individuals in columns. Note that dosages must be in converted form, i.e. after having run the convert_marker_dosages function. Errors may result otherwise.
dosage_matrix.orig	Optional, by default NULL.The unconverted dosages (i.e. raw dosage data before using the convert_marker_dosages function). Required if original_coding is TRUE.
remove_markers	Optional vector of marker names to remove from the maps. Default is NULL.
N_linkages	Number of significant linkages (as defined in homologue_lg_assignment) required for high-confidence linkage group assignment.
lower_bound	Numeric. Lower bound for the rate at which homologue linkages (fraction of total for that marker) are recognised.
ploidy	Integer. Ploidy of the organism.
ploidy2	Optional integer, by default NULL. Ploidy of parent 2, if different from parent 1.
marker_assignment.1	A marker assignment matrix for parent 1 with markernames as rownames and at least containing the column "Assigned_LG".
marker_assignment.2	A marker assignment matrix for parent 2 with markernames as rownames and at least containing the column "Assigned_LG".
parent1	character vector with names of the samples of parent 1
parent2	character vector with names of the samples of parent 2
original_coding	Logical. Should the phased map use the unconverted dosage coding or not?
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 unphased markers be recorded?

Examples

```
data("integrated.maplist", "screened_data3", "marker_assignments_P1","marker_assignments_P2")
create_phased_maplist(integrated.maplist,
                      dosage_matrix.conv = screened_data3,
                      marker_assignment.1=marker_assignments_P1,
                      marker_assignment.2=marker_assignments_P2)
```

define_LG_structure *Generate linkage group and homologue structure of SxN markers*

Description

Function which organises the output of `cluster_SN_markers` into a data frame of numbered linkage groups and homologues. Only use this function if it is clear from the graphical output of `cluster_SN_markers` that there are LOD scores present which define both chromosomes (lower LOD) and homologues (higher LOD).

Usage

```
define_LG_structure(cluster_list, LOD_chm, LOD_hom, LG_number, log = NULL)
```

Arguments

`cluster_list`	A list of cluster_stacks, the output of `cluster_SN_markers`.
`LOD_chm`	Integer. The LOD threshold specifying at which LOD score the markers divide into chromosomal groups
`LOD_hom`	Integer. The LOD threshold specifying at which LOD score the markers divide into homologue groups
`LG_number`	Integer. Expected number of chromosomes (linkage groups). Note that if this number of clusters are not present at LOD_chm, the function will abort.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A data.frame with markers classified by homologue and linkage group.

Examples

```
data("P1_homologues")
ChHomDf<-define_LG_structure(cluster_list=P1_homologues,LOD_chm=3.5,LOD_hom=5,LG_number=5)
```

finish_linkage_analysis *Linkage analysis between all markertypes within LG.*

Description

`finish_linkage_analysis` is a wrapper for `linkage`. Performs linkage calculations between all markertypes within a linkage group.

Usage

```
finish_linkage_analysis(marker_assignment, dosage_matrix,
  marker_combinations = NULL, target_parent = "P1", other_parent = "P2",
  convert_palindrome_markers = TRUE, ploidy = 4, ploidy2 = NULL,
  pairing = c("random", "preferential"), prefPars = c(0, 0), LG_number,
  verbose = TRUE, log = NULL, ...)
```

Arguments

marker_assignment	A marker assignment matrix with markernames as rownames and at least containing the column "Assigned_LG".
dosage_matrix	An integer matrix with markers in rows and individuals in columns.
marker_combinations	A matrix with four columns specifying marker combinations to calculate linkage. If NULL all combinations are used for which there are rf functions. Dosages of markers should be in the same order as specified in the names of rf functions. E.g. if using 1.0_2.0 and 1.0_3.0 types use: `matrix(c(1,0,2,0,1,0,3,0), byrow = TRUE, ncol = 4)`
target_parent	Character string specifying target parent.
other_parent	Character string specifying other parent.
convert_palindrome_markers	Logical. Should markers that behave the same for both parents be converted to a workable format for that parent? E.g.: should 3.1 markers be converted to 1.3?
ploidy	Ploidy level of parent 1.
ploidy2	Integer, by default NULL. If parental ploidies differ, the ploidy of parent 2.
pairing	Type of pairing at meiosis, with options "random" or "preferential".
prefPars	The estimates for preferential pairing parameters for parent 1 and 2, in range 0 <= p < 2/3. By default this is c(0,0) (so, no preferential pairing). See the function test_prefpairing and the vignette for more details.
LG_number	Number of linkage groups (chromosomes).
verbose	Should messages be send to stdout or log?
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
...	(Other) arguments passed to linkage

Value

Returns a matrix with marker assignments. Number of linkages of 1.0 markers are artificial.

Examples

```
## Not run:
data("screened_data3", "marker_assignments_P1")
linkages_list_P1<-finish_linkage_analysis(marker_assignment=marker_assignments_P1,
                                          dosage_matrix=screened_data3,
```

```
                                        target_parent="P1",
                                        other_parent="P2",
                                        convert_palindrome_markers=FALSE,
                                        ploidy=4,
                                        pairing="random",
                                        LG_number=5)

## End(Not run)
```

`get_markertype_combinations` *Visualize and get all markertype combinations for which there are functions in polymapR*

Description

Visualize and get all markertype combinations for which there are functions in polymapR

Usage

```
get_markertype_combinations(ploidy, pairing, nonavailable_combinations = TRUE)
```

Arguments

`ploidy`	Ploidy level
`pairing`	Type of pairing. Either "random" or "preferential".
`nonavailable_combinations`	Logical. Should nonavailable combinations be plotted with grey lines?

Value

A matrix with two columns. Each row represents a function with the first and second markertype.

Examples

```
get_markertype_combinations(4, "random")
```

homologue_lg_assignment *Assign markers to linkage groups and homologues.*

Description

This is a wrapper combining linkage and assign_linkage_group. It is used to assign all marker types to linkage groups by using linkage information with 1.0 markers. It allows for input of marker assignments for which this analysis has already been performed.

Usage

```
homologue_lg_assignment(dosage_matrix, assigned_list, assigned_markertypes,
  SN_functions = NULL, LG_hom_stack, target_parent = "P1",
  other_parent = "P2", convert_palindrome_markers = TRUE, ploidy = 4,
  ploidy2 = NULL, pairing = "random", LG_number = 5, LOD_threshold = 3,
  write_intermediate_files = TRUE, log = NULL, ...)
```

Arguments

dosage_matrix	A dosage matrix.
assigned_list	List of data.frames with marker assignments for which the assignment analysis is already performed.
assigned_markertypes	List of integer vectors of length 2. Specifying the markertypes in the same order as assigned_list.
SN_functions	A vector of function names to be used. If NULL all remaining linkage functions with SN markers are used.
LG_hom_stack	A data.frame with markernames ("SxN_Marker"), linkage group ("LG") and homologue ("homologue")
target_parent	A character string specifying the target parent.
other_parent	A character string specifying the other parent.
convert_palindrome_markers	Logical. Should markers that behave the same for both parents be converted to a workable format for that parent? E.g.: should 3.1 markers be converted to 1.3?
ploidy	Ploidy level of parent 1.
ploidy2	Integer, by default NULL. If parental ploidies differ, the ploidy of parent 2.
pairing	Type of pairing. Either "random" or "preferential".
LG_number	Expected number of chromosomes (linkage groups).
LOD_threshold	LOD threshold at which a linkage is considered significant.
write_intermediate_files	Logical. Write intermediate linkage files to working directory?
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
...	Arguments passed to linkage

Value

A data.frame specifying marker assignments to linkage group and homologue.

Examples

```
## Not run:
data("screened_data3", "P1_SxS_Assigned", "P1_DxN_Assigned", "LGHomDf_P1_1")
Assigned_markers<-homologue_lg_assignment(screened_data3,
                         assigned_list = list(P1_SxS_Assigned, P1_DxN_Assigned),
                         assigned_markertypes = list(c(1,1), c(2,0)),
                         LG_hom_stack = LGHomDf_P1_1,
                         write_intermediate_files=FALSE)

## End(Not run)
```

---

integrated.maplist *A nested list with integrated maps*

Description

A nested list with integrated maps

Usage

```
integrated.maplist
```

Format

An object of class list of length 5.

LGHomDf_P1_1 *A data.frame specifying the assigned homologue and linkage group number per SxN marker*

Description

A data.frame specifying the assigned homologue and linkage group number per SxN marker

Usage

```
LGHomDf_P1_1

LGHomDf_P2_1

LGHomDf_P2_2
```

Format

- SxN_Marker. Markername of simplex nulliplex marker
- homologue. Assigned homologue number
- LG Assigned. linkage group number

`linkage` *Calculate recombination frequency, LOD and phase*

Description

`linkage` is used to calculate recombination frequency, LOD and phase within one type of marker or between two types of markers.

Usage

```
linkage(dosage_matrix, markertype1 = c(1, 0), markertype2 = NULL,
  target_parent = "P1", other_parent = "P2", G2_test = FALSE,
  convert_palindrome_markers = TRUE, LOD_threshold = 0, ploidy = c(4, 6),
  ploidy2 = NULL, pairing = c("random", "preferential"), prefPars = c(0,
  0), combinations_per_iter = NULL, verbose = TRUE, full_output = FALSE,
  iter_RAM = 500, ncores = 1, log = NULL)
```

Arguments

`dosage_matrix`	An integer matrix with markers in rows and individuals in columns.
`markertype1`	A vector of length 2 specifying the first markertype to compare. The first element specifies the dosage in `target_parent`, the second in `other_parent`.
`markertype2`	A vector of length 2 specifying the first markertype to compare. This argument is optional. If not specified, the funciton will calculate linkage within the markertype as specified by `markertype1`. The first element specifies the dosage in `target_parent`, the second in `other_parent`.
`target_parent`	Character string specifying the target parent as provided in the columnnames of dosage_matrix
`other_parent`	Character string specifying the other parent as provided in the columnnames of dosage_matrix
`G2_test`	Apply a G2 test (LOD of indepedence) in addition to the LOD of linkage.
`convert_palindrome_markers`	Logical. Should markers that behave the same for both parents be converted to a workable format for that parent? E.g.: should 3.1 markers be converted to 1.3? If unsure, set to TRUE.
`LOD_threshold`	Minimum LOD score of linkages to report. Recommended to use for large number (> millions) of marker comparisons in order to reduce memory usage.
`ploidy`	Integer. The ploidy of parent 1.

ploidy2	Integer, by default NULL. If parental ploidies differ, the ploidy of parent 2.
pairing	Type of pairing. "random" or "preferential".
prefPars	The estimates for preferential pairing parameters for parent 1 and 2, in range 0 <= p < 2/3. By default this is c(0,0) (so, no preferential pairing). See the function test_prefpairing and the vignette for more details.
combinations_per_iter	Optional integer. Number of marker combinations per iteration.
verbose	Should messages be send to stdout?
full_output	Logical, by default FALSE. If TRUE, the complete output over all phases and showing marker combination counts is returned.
iter_RAM	A (very) conservative estimate of working memory in megabytes used per core. It only takes the size frequency matrices into account. Actual usage is more, espacially with large number of linkages that are reported. Reduce memory usage by using a higher LOD_threshold.
ncores	Number of cores to use. Works both for Windows 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.

Value

Returns a data.frame with columns:

marker_a	first marker of comparison. If markertype2 is specified, it has the type of markertype1.
marker_b	second marker of comparison. It has the type of markertype2 if specified.
r	(estimated) recombinations frequency
LOD	(estimated) LOD score
phase	phase between markers

Examples

```
data("screened_data3")
SN_SN_P1 <- linkage(dosage_matrix = screened_data3,
                    markertype1 = c(1,0),
                    target_parent = "P1",
                    other_parent = "P2",
                    ploidy = 4,
                    pairing = "random",
                    ncores = 1
                    )
```

maplist_P1 *A list of maps of one parent*

Description

A list of maps of one parent

Usage

```
maplist_P1

maplist_P1_subset

maplist_P2_subset
```

Format

An object of class `list` of length 5.

marker_binning *Perform binning of markers.*

Description

`marker_binning` allows for binning of very closely linked markers and choses one representative.

Usage

```
marker_binning(dosage_matrix, linkage_df, r_thresh = NA, lod_thresh = NA,
  target_parent = "P1", other_parent = "P2", max_marker_nr = NULL,
  max_iter = 10, log = NULL)
```

Arguments

`dosage_matrix`	A dosage `matrix`.
`linkage_df`	A linkage `data.frame`.
`r_thresh`	Numeric. Threshold at which markers are binned. Is calculated if NA.
`lod_thresh`	Numeric. Threshold at which markers are binned. Is calculated if NA.
`target_parent`	A character string specifying the name of the target parent.
`other_parent`	A character string specifying the name of the other parent.
`max_marker_nr`	The maximum number of markers per homologue. If specified, LOD threshold is optimized based on this number.
`max_iter`	Maximum number of iterations to find optimum LOD threshold. Only used if `max_marker_nr` is specified.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A list with the following components:

binned_df	A linkage data.frame with binned markers removed.
removed	A data.frame containing binned markers and their representatives.
left	Integer. Number markers left.

Examples

```
data("screened_data3", "all_linkages_list_P1_split")
binned_markers<-marker_binning(screened_data3, all_linkages_list_P1_split[["LG2"]][["homologue3"]])
```

---

marker_binning_list *Bin markers that are in a nested list*

Description

This is a wrapper for marker_binning. It applies marker_binning to a nested list as output of split_linkage_info

Usage

```
marker_binning_list(dosage_matrix, linkage_list, r_thresh = NA,
  lod_thresh = NA, return_removed_marker_info = FALSE, log = NULL, ...)
```

Arguments

dosage_matrix	A dosage matrix
linkage_list	A nested list with linkage group on the first level and homologue on the second.
r_thresh	Numeric. Threshold at which markers are binned. Is calculated if NA.
lod_thresh	Numeric. Threshold at which markers are binned. Is calculated if NA.
return_removed_marker_info	Logical. Should removed marker information be returned? If TRUE, output is a list containing the linkage list with binned markers removed and a dataframe with removed marker information.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
...	Arguments passed to marker_binning

Value

A list of linkage data.frames with binned markers removed. If return_removed_marker_info = TRUE, a list containing the above linkage dataframes (linkage_list) and a dataframe with removed marker information, called removed_markers.

Examples

```
data("screened_data3", "all_linkages_list_P1_split")
mb<-marker_binning_list(screened_data3,
                    all_linkages_list_P1_split,
                    target_parent="P1",
                    other_parent="P2")
```

`marker_data_summary` *Summarize marker data*

Description

Gives a frequency table of different markertypes, relative frequency per markertype of incompatible offspring and the names of incompatible progeny.

Usage

```
marker_data_summary(dosage_matrix, ploidy = 4, pairing = c("random",
  "preferential"), parent1 = "P1", parent2 = "P2",
  progeny_incompat_cutoff = 0.1, verbose = TRUE, log = NULL)
```

Arguments

`dosage_matrix`	An integer matrix with markers in rows and individuals in columns.
`ploidy`	Integer. Ploidy of plant species.
`pairing`	Type of pairing. "random" or "preferential".
`parent1`	Name of first parent. Usually maternal parent.
`parent2`	Name of second parent. Usually paternal parent.
`progeny_incompat_cutoff`	The relative number of incompatible dosages per genotype that results in reporting this genotype as incompatible. Incompatible dosages are greater than maximum number of alleles than can be inherited or smaller than the minimum number of alleles that can be inherited.
`verbose`	Logical, by default `TRUE` - should intermediate messages be written to stout?
`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 containing the following components:

`parental_info`	frequency table of different markertypes. Names start with parentnames, and behind that the dosage score.
`offspring_incompatible`	relative frequency of incompatible offspring with same layout as parental_info.
`progeny_incompatible`	progeny names having incompatible dosage scores higher than threshold at progeny_incompat_cutoff.

Examples

```
data("ALL_dosages")
summary_list<-marker_data_summary(dosage_matrix = ALL_dosages)
```

`MDSMap_from_list` *Wrapper function for MDSMap to generate linkage maps from list of pairwise linkage estimates*

Description

Create multidimensional scaling maps from a list of linkages

Usage

```
MDSMap_from_list(linkage_list, write_to_file = FALSE,
  mapdir = "mapping_files_MDSMap", plot_prefix = "", log = NULL, ...)
```

Arguments

`linkage_list`	A named `list` with r and LOD of markers within linkage groups.
`write_to_file`	Should output be written to a file? By default `FALSE`, if `TRUE` then output, including plots from `MDSMap` are saved in the same directory as the one used for input files. These plots are currently saved as pnf images. If a different plot format is required (e.g. for publications), then run the `MDSMap` function `estimate.map` (or similar) directly and save the output with a different plotting function as wrapper around the map function call.
`mapdir`	Directory to which map input files are initially written. Also used for output if `write_to_file=TRUE`
`plot_prefix`	prefix for the filenames of output plots.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
`...`	Arguments passed to `estimate.map`.

Examples

```
## Not run:
data("all_linkages_list_P1")
maplist_P1 <- MDSMap_from_list(all_linkages_list_P1[1])

## End(Not run)
```

merge_homologues *Merge homologues*

Description

Based on additional information, homologue fragments, separated during clustered should be merged again. merge_homologues allows to merge homologues per linkage group based on user input.

Usage

```
merge_homologues(LG_hom_stack, ploidy, linkage_group, mergeList = NULL,
  log = NULL)
```

Arguments

LG_hom_stack	A data.frame with markernames, linkage group ("LG") and homologue ("homologue")
ploidy	The ploidy level of the plant species.
linkage_group	The linkage group where the to be merged homologue fragments are in.
mergeList	A list of vectors of length 2, specifying the numbers of the homologue fragments to be merged. User input is asked if NULL.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A modified LG_hom_stack

Examples

```
data("LGHomDf_P2_1")
merged<-merge_homologues(LGHomDf_P2_1, 4, 2, list(c(1,5)))
```

merge_marker_assignments *Merge marker assignments*

Description

merge_marker_assignments Merges 1.0 backbone object with marker assignment objects

Usage

```
merge_marker_assignments(dosage_matrix, target_parent = "P1",
  other_parent = "P2", LG_hom_stack, SN_linked_markers, ploidy,
  LG_number = 5, log = NULL)
```

Arguments

dosage_matrix	A dosage matrix.
target_parent	Character string specifying target parent.
other_parent	Character string specifying other parent.
LG_hom_stack	data.frame specifying 1.0 marker assignments to linkage groups and homologues.
SN_linked_markers	a list of marker assignment objects
ploidy	Ploidy level of plant species.
LG_number	Number of linkage groups (chromosomes).
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

Returns a matrix with marker assignments. Number of linkages of 1.0 markers are artificial.

Examples

```
data("screened_data3", "LGHomDf_P1_1", "P1_SxS_Assigned", "P1_DxN_Assigned")
merged_assignment<-merge_marker_assignments(screened_data3, target_parent="P1",
                        other_parent="P2",
                        LG_hom_stack=LGHomDf_P1_1,
                        SN_linked_markers=list(P1_SxS_Assigned, P1_DxN_Assigned),
                        ploidy=4,
                        LG_number=5)
```

---

orient_and_merge_maps *Align and integrate maps*

Description

Align homologues to the same orientation and integrates maps using LPmerge

Usage

```
orient_and_merge_maps(maplist_P1, maplist_P2, parent_ID_1 = "P1",
  parent_ID_2 = "P2", connection_threshold = 2, LPmerge_interval = 4,
  single_LPmerge = FALSE, plot_graph = FALSE, ploidy = 4, log = NULL)
```

Arguments

maplist_P1, maplist_P2	A list of length=ploidy, with data.frames containing markernames and position.
parent_ID_1, parent_ID_2	Character string with parent IDs
connection_threshold	The number of markers two homologues should have in common to have a significant connection
LPmerge_interval	The max.interval value used for LPmerge
single_LPmerge	Logical, by default FALSE. If TRUE then only a single merge with max.interval = LPmerge_interval will be run, otherwise all intervals up to LPmerge_interval will also be tested.
plot_graph	Should the connection network be drawn?
ploidy	The ploidy of the organism
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Examples

```
data("maplist_P1_subset")
data("maplist_P2_subset")
## Not run:
## Example temporarily suspended (April 2018): LPmerge CRAN issues
integrated_map_LG2<-orient_and_merge_maps(maplist_P1=maplist_P1_subset[["LG4"]],
                                          maplist_P2=maplist_P2_subset[["LG4"]],
                                          plot_graph = TRUE)

## End(Not run)
```

---

overviewSNlinks *Plotting 1.0 links between homologues*

Description

overviewSNlinks is written to enable merging of homologue fractions. Fractions of homologues will have more markers in coupling than in repulsion, whereas separate homologues will only have markers in repulsion.

Usage

```
overviewSNlinks(linkage_df, LG_hom_stack, LG_number, LOD_threshold,
  ymax = NULL, log = NULL)
```

Arguments

linkage_df	A data.frame as output of linkage with arguments markertype1=c(1,0) and markertype2=NULL.
LG_hom_stack	A data.frame with a column "SxN_Marker" specifying markernames, a column "homologue" specifying homologue cluster and "LG" specifying linkage group.
LG_number	Integer. Chromosome (linkage group) number.
LOD_threshold	Numeric. LOD threshold of linkages which are plotted.
ymax	Maximum y-limit of the plots.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Examples

```
data("SN_SN_P1", "LGHomDf_P1_1")
overviewSNlinks(linkage_df=SN_SN_P1,
                LG_hom_stack=LGHomDf_P1_1,
                LG_number=5,
                LOD_threshold=3)
```

P1_homologues *A list of cluster stacks at different LOD scores*

Description

A list of cluster stacks at different LOD scores

Usage

```
P1_homologues

P2_homologues

P2_homologues_triploid
```

Format

A list with with LOD thresholds as names. The list contains dataframes with the following format:

- marker. markername
- pseudohomologue. name of (pseudo)homologue

P1_SxS_Assigned *A* data.frame *with marker assignments*

Description

A data.frame with marker assignments

Usage

```
P1_SxS_Assigned

P2_SxS_Assigned

P2_SxS_Assigned_2

P1_DxN_Assigned

P2_DxN_Assigned

marker_assignments_P1

marker_assignments_P2
```

Format

A data.frame with at least the following columns:

- Assigned_LG. The assigned linkage group
- Assigend_hom1. The homologue with most linkages

The columns LG1 - LGn and Hom1 - Homn give the number of hits per marker for that linkage group/homologue. Assigned_hom2 .. gives the nth homologue with most linkages.

p4_functions *Calculate recombination frequency, LOD and log-likelihood from frequency tables in a preferential pairing tetraploid*

Description

This group of functions is called by linkage.

Arguments

x	A frequency table of the different classes of dosages in the progeny. The column names start with "n_". Followed by the dosage of the first marker and then of the second.
p1	Preferential pairing parameter for parent 1, numeric value in range 0 <= p1 < 2/3
p2	Preferential pairing parameter for parent 2, numeric value in range 0 <= p2 < 2/3
ncores	Number of cores to use for parallel processing (deprecated).

Value

A list with the following items:

r_mat	A matrix with recombination frequencies for the different phases
LOD_mat	A matrix with LOD scores for the different phases
logL_mat	A matrix with log likelihood ratios for the different phases
phasing_strategy	A character string specifying the phasing strategy. "MLL" for maximum likelihood en "MINR" for minimum recombination frequency.
possible_phases	The phases between markers that are possible. Same order and length as column names of output matrices.

---

parental_quantities *Calculate frequency of each markertype.*

Description

Plots and returns frequency information for each markertype.

Usage

```
parental_quantities(dosage_matrix, parent1 = "P1", parent2 = "P2",
  log = NULL, ...)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns.
parent1	Character string specifying the first (usually maternal) parentname.
parent2	Character string specifying the second (usually paternal) parentname.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
...	Arguments passed to barplot

Value

A named vector containing the frequency of each markertype in the dataset.

Examples

```
data("ALL_dosages")
data("screened_data")
parental_quantities(dosage_matrix=ALL_dosages)
parental_quantities(dosage_matrix=screened_data)
```

PCA_progeny *Perform a PCA on progeny*

Description

Principal component analysis in order to identify individuals that deviate from the population.

Usage

```
PCA_progeny(dosage_matrix, highlight = NULL, colors = NULL, log = NULL)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns.
highlight	A list of character vectors specifying individual names that should be highlighted
colors	Highlight colors. Vector of the same length as highlight.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Details

Missing values are imputed by taking the mean of marker dosages per marker.

Examples

```
data("ALL_dosages")
PCA_progeny(dosage_matrix=ALL_dosages, highlight=list(c("P1", "P2")), colors="red")
```

phased.maplist *A list of phased maps*

Description

A list of phased maps

Usage

```
phased.maplist
```

Format

An object of class `list` of length 5.

phase_SN_diploid *Phase 1.0 markers at the diploid level*

Description

`phase_SN_diploid` phases simplex x nulliplex markers for a diploid parent.

Usage

```
phase_SN_diploid(linkage_df, cluster_list, LOD_chm = 3.5, LG_number = 3,
  independence_LOD = FALSE, log = NULL)
```

Arguments

`linkage_df`	A linkage data.frame as output of `linkage` calculating linkage between 1.0 markers.
`cluster_list`	A list of cluster_stacks, the output of `cluster_SN_markers`.
`LOD_chm`	Integer. The LOD threshold specifying at which LOD score the markers divide into chromosomal groups
`LG_number`	Expected number of chromosomes (linkage groups)
`independence_LOD`	Logical. Should the LOD of independence be used for clustering? (by default, `FALSE`.)
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout (console).

Value

A data.frame with markers classified by homologue and linkage group.

Examples

```
data("SN_SN_P2_triploid","P2_homologues_triploid")
cluster_list2<-phase_SN_diploid(SN_SN_P2_triploid,P2_homologues_triploid,LOD_chm=5)
```

`plot_hom_vs_LG` *Plot homologue position versus integrated positions*

Description

Plot homologue position versus integrated positions

Usage

```
plot_hom_vs_LG(map_df, maplist_homologue)
```

Arguments

`map_df`	A dataframe of a map that defines a linkage group.
`maplist_homologue`	A list of maps were each item represents a homoloogue.

Examples

```
data("integrated.maplist", "maplist_P1_subset")
colnames(integrated.maplist[["LG2"]]) <- c("marker", "position", "QTL_LOD")
plot_hom_vs_LG(map_df = integrated.maplist[["LG2"]],
               maplist_homologue = maplist_P1_subset[["LG2"]])
```

`plot_linkage_df` *Plot r versus LOD grouped by phase*

Description

`plot_linkage_df` plots r versus LOD, colour separated for different phases.

Usage

```
plot_linkage_df(linkage_df, r_max = 0.5, ...)
```

Arguments

`linkage_df`	A linkage data.frame as output of `linkage`.
`r_max`	Maximum r value to plot
`...`	Arguments passed to base plot function

Examples

```
data("SN_SN_P1")
plot_linkage_df(SN_SN_P1)
```

`plot_map` *Plot linkage maps*

Description

Makes a simple plot of a list of generated linkage maps

Usage

```
plot_map(maplist, highlight = NULL, bg_col = "grey",
  highlight_col = "yellow", colname_in_mark = NULL,
  colname_beside_mark = NULL, palette_in_mark = colorRampPalette(c("white",
  "purple")), palette_beside_mark = colorRampPalette(c("white", "green")),
  color_by_type = FALSE, dosage_matrix = NULL, legend = FALSE,
  legend.x = 1, legend.y = 120, ...)
```

Arguments

`maplist`	A list of maps. In the first column marker names and in the second their position.
`highlight`	A list of the same length of maplist with vectors of length 2 that specifies the limits in cM from and to which the plotted chromosomes should be highlighted.
`bg_col`	The background colour of the map.
`highlight_col`	The color of the highlight. Only used if `highlight` is specified.
`colname_in_mark`	Optional. The column name of the value to be plotted as marker color.
`colname_beside_mark`	Optional. The column name of the value to be plotted beside the markers.
`palette_in_mark, palette_beside_mark`	Color palette used to plot values. Only used if colnames of the values are specified.
`color_by_type`	Logical. Should the markers be coloured by type? If TRUE, dosage_matrix should be specified.
`dosage_matrix`	Optional (by default `NULL`). Dosage matrix of marker genotypes, input of `linkage`
`legend`	Logical. Should a legend be drawn?
`legend.x`	Optional. The x value of the coordinates of the legend.
`legend.y`	Optional. The y value of the coordinates of the legend.
`...`	Arguments passed to `plot`

Examples

```
data("maplist_P1")
plot_map(maplist = maplist_P1, colname_in_mark = "nnfit", bg_col = "white",
         palette_in_mark = colorRampPalette(c("blue", "purple", "red")),
         highlight = list(c(20, 60),
         c(60,80),
         c(20,30),
         c(40,70),
         c(60,80)))
```

`plot_phased_maplist` *Visualise the phased homologue maplist*

Description

`plot_phased_maplist` is a function for visualising a phased maplist, the output of `create_phased_maplist`

Usage

```
plot_phased_maplist(phased.maplist, ploidy = 4, ploidy2 = NULL,
  cols = c("black", "darkred", "navyblue"), width = 0.2, mapTitles = NULL)
```

Arguments

`phased.maplist`	A list of phased linkage maps, the output of `create_phased_maplist`
`ploidy`	Integer. Ploidy of the organism.
`ploidy2`	Optional integer, by default `NULL`. Ploidy of parent 2, if different from parent 1.
`cols`	Vector of colours for the integrated, parent1 and parent2 maps, respectively.
`width`	Width of the linkage maps, by default 0.2
`mapTitles`	Optional vector of titles for maps, by default names of maplist, or titles LG1, LG2 etc. are used.

Examples

```
data("phased.maplist")
plot_phased_maplist(phased.maplist)
```

`polymapR` *Linkage analysis in polyploids*

Description

This package uses dosage-scored SNP markers from an F1 cross to perform linkage analysis in polyploids

r3_functions — *Calculate recombination frequency, LOD and log-likelihood from frequency tables in a random pairing triploid from a tetraploid x diploid cross.*

Description

This group of functions is called by `linkage`.

Arguments

`x`	A frequency table of the different classes of dosages in the progeny. The column names start with `"n_"`. Followed by the dosage of the first marker and then of the second.
`ncores`	Number of cores to use for parallel processing (deprecated).

Value

A list with the following items:

`r_mat`	A matrix with recombination frequencies for the different phases
`LOD_mat`	A matrix with LOD scores for the different phases
`logL_mat`	A matrix with log likelihood ratios for the different phases
`phasing_strategy`	A character string specifying the phasing strategy. `"MLL"` for maximum likelihood en `"MINR"` for minimum recombination frequency.
`possible_phases`	The phases between markers that are possible. Same order and length as column names of output matrices.

r4_functions — *Calculate recombination frequency, LOD and log-likelihood from frequency tables in a random pairing tetraploid*

Description

This group of functions is called by `linkage`.

Arguments

`x`	A frequency table of the different classes of dosages in the progeny. The column names start with `"n_"`. Followed by the dosage of the first marker and then of the second.
`ncores`	Number of cores to use for parallel processing (deprecated).

Value

A list with the following items:

r_mat	A matrix with recombination frequencies for the different phases
LOD_mat	A matrix with LOD scores for the different phases
logL_mat	A matrix with log likelihood ratios for the different phases
phasing_strategy	A character string specifying the phasing strategy. "MLL" for maximum likelihood en "MINR" for minimum recombination frequency.
possible_phases	The phases between markers that are possible. Same order and length as column names of output matrices.

---

r6_functions *Calculate recombination frequency, LOD and log-likelihood from frequency tables in a random pairing hexaploid*

Description

This group of functions is called by linkage.

Arguments

x	A frequency table of the different classes of dosages in the progeny. The column names start with "n_". Followed by the dosage of the first marker and then of the second.

Value

A list with the following items:

r_mat	A matrix with recombination frequencies for the different phases
LOD_mat	A matrix with LOD scores for the different phases
logL_mat	A matrix with log likelihood ratios for the different phases
phasing_strategy	A character string specifying the phasing strategy. "MLL" for maximum likelihood en "MINR" for minimum recombination frequency.
possible_phases	The phases between markers that are possible. Same order and length as column names of output matrices.

r_LOD_plot *Plot r versus LOD*

Description

r_LOD_plot plots r versus LOD, colour separated for different phases.

Usage

```
r_LOD_plot(linkage_df, plot_main = "", chm = NA, r_max = 0.5)
```

Arguments

linkage_df	A linkage data.frame as output of linkage.
plot_main	A character string specifying the main title
chm	Integer specifying chromosome
r_max	Maximum r value to plot

Examples

```
data("SN_SN_P1")
r_LOD_plot(SN_SN_P1)
```

screen_for_duplicate_individuals *Screen for duplicate individuals*

Description

screen_for_duplicate_individuals identifies and merges duplicate individuals.

Usage

```
screen_for_duplicate_individuals(dosage_matrix, cutoff = NULL, plot_cor = T,
  log = NULL)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns.
cutoff	Correlation coefficient cut off. At this correlation coefficient, individuals are merged. If NULL user input will be asked after plotting.
plot_cor	Logical. Should correlation coefficients be plotted? Can be memory/CPU intensive with high number of individuals.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A matrix similar to dosage_matrix, with merged duplicate individuals.

Examples

```
data("segregating_data")
dupscreened<-screen_for_duplicate_individuals(dosage_matrix=segregating_data,
                                              cutoff=0.9,
                                              plot_cor=TRUE)
## Not run:
#user input:
data("segregating_data")
screen_for_duplicate_individuals(dosage_matrix=segregating_data, plot_cor=TRUE)

## End(Not run)
```

---

screen_for_duplicate_markers

Screen for and remove duplicated markers

Description

`screen_for_duplicate_markers` identifies and merges duplicate markers.

Usage

```
screen_for_duplicate_markers(dosage_matrix, merge_NA = TRUE,
  plot_cluster_size = TRUE, log = NULL)
```

Arguments

`dosage_matrix`	An integer matrix with markers in rows and individuals in columns.
`merge_NA`	Logical. Should missing values be imputed if non-NA in duplicated marker? By default, `TRUE`. If `FALSE` the dosage scores of representing marker are represented in the filtered_dosage_matrix.
`plot_cluster_size`	Logical. Should an informative plot about duplicate cluster size be given? By default, `TRUE`.
`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:

- bin_list list of binned markers. The list names are the representing markers. This information can later be used to enrich the map with binned markers.
- filtered_dosage_matrix dosage_matrix with merged duplicated markers. The markers will be given the name of the marker with least missing values.

Examples

```
data("screened_data3")
dupmscreened <- screen_for_duplicate_markers(screened_data3)
```

`screen_for_NA_values` *Screen marker data for NA values*

Description

`screen_for_NA_values` identifies and can remove rows or columns of a marker dataset based on the relative frequency of missing values.

Usage

```
screen_for_NA_values(dosage_matrix, margin = 1, cutoff = NULL,
  parentnames = c("P1", "P2"), plot_breakdown = FALSE, log = NULL,
  print.removed = TRUE)
```

Arguments

`dosage_matrix`	An integer matrix with markers in rows and individuals in columns.
`margin`	An integer at which margin the missing value frequency will be calculated. A value of 1 means rows (markers), 2 means columns (individuals)
`cutoff`	Missing value frequency cut off. At this frequency, rows or columns are removed from the dataset. If NULL user input will be asked after plotting the missing value frequency histogram.
`parentnames`	A character vector of length 2, specifying the parent names.
`plot_breakdown`	Logical. Should the percentage of markers removed as breakdown per marker-type be plotted? Can only be used if margin = 1.
`log`	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.
`print.removed`	Logical. Should removed instances be printed?

Value

A matrix similar to dosage_matrix, with rows or columns removed that had a higher missing value frequency than specified.

Examples

```
data("segregating_data")
data("screened_data")
screened_markers<-screen_for_NA_values(dosage_matrix=segregating_data, margin=1, cutoff=0.1)
screened_indiv<-screen_for_NA_values(dosage_matrix=screened_data, margin=2, cutoff=0.1)
## Not run:
#user input:
```

```
#screen_for_NA_values(dosage_matrix=segregating_data, margin=1, cutoff=NULL)

## End(Not run)
```

SNSN_LOD_deviations *Identify deviations in LOD scores between pairs of simplex x nulliplex markers*

Description

SNSN_LOD_deviations checks whether the LOD scores obtained in the case of pairs of simplex x nulliple markers are compatible with expectation. This can help identify problematic linkage estimates which can adversely affect marker clustering.

Usage

```
SNSN_LOD_deviations(linkage_df, ploidy, N, plot_expected = TRUE,
  alpha = c(0.05, 0.2), phase = c("coupling", "repulsion"))
```

Arguments

linkage_df	A linkage data.frame as output of linkage.
ploidy	Integer. The ploidy level of the species.
N	Numeric. The number of F1 individuals in the mapping population.
plot_expected	Logical. Plot the observed and expected relationship between r and LOD.
alpha	Numeric. Vector of upper and lower tolerances around expected line.
phase	Character string. Specify which phase to examine for deviations (usually this is "coupling" phase).

Value

A vector of deviations in LOD scores outside the range defined by tolerances input alpha

Examples

```
data("SN_SN_P1")
SNSN_LOD_deviations(SN_SN_P1,4,198)
```

SN_SN_P1 *A linkage* data.frame.

Description

A linkage data.frame.

Usage

```
SN_SN_P1

SN_SN_P2

SN_SS_P1

SN_SS_P2

SN_DN_P1

SN_DN_P2

SN_SN_P2_triploid
```

Format

- marker_a. First marker in comparison
- marker_b. Second marker in comparison
- r. recombination frequency
- LOD. LOD score
- phase. The phase between markers

split_linkage_info *Split linkage information into homologues*

Description

After running finish_linkage_analysis recombination frequency, LOD and phase are grouped by linkage group. This functions classifies them into homologue within the linkage group.

Usage

```
split_linkage_info(all_linkages, marker_assignment, ploidy, log = NULL)
```

Arguments

all_linkages	A list of linkage information divided by linkage group as output of finish_linkage_analysis.
marker_assignment	A complete marker assignment data.frame containing all markers.
ploidy	Integer. The ploidy level of the plant species.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Value

A nested list with linkage group on the first level and homologue on the second.

Examples

```
data("marker_assignments_P1", "all_linkages_list_P1")
splitted_list<-split_linkage_info(all_linkages_list_P1, marker_assignments_P1, 4)
```

---

test_prefpairing *Check for and estimate preferential pairing*

Description

Identify closely-mapped repulsion-phase simplex x nulliplex markers and test these for preferential pairing, including estimating a preferential pairing parameter.

Usage

```
test_prefpairing(dosage_matrix, maplist, LG_hom_stack, target_parent = "P1",
  other_parent = "P2", ploidy, min_cM = 0.5, adj.method = "fdr",
  verbose = TRUE)
```

Arguments

dosage_matrix	An integer matrix with markers in rows and individuals in columns.
maplist	A list of integrated chromosomal maps, as generated by e.g. MDSMap_from_list. In the first column marker names and in the second their position.
LG_hom_stack	A data.frame with markernames ("SxN_Marker"), linkage group ("LG") and homologue ("homologue"), the output of define_LG_structure or bridgeHomologues usually.
target_parent	Character string specifying the parent to be tested for preferential pairing as provided in the columnnames of dosage_matrix, by default "P1".
other_parent	The other parent, by default "P2"
ploidy	The ploidy level of the species, by default 4 (tetraploid) is assumed.
min_cM	The smallest distance to be considered a true distance on the linkage map, by default distances less than 0.5 cM are considered essentially zero.

adj.method	Method to correct p values of Binomial test for multiple testing, by default the FDR correction is used, other options are available, inherited from p.adjust
verbose	Should messages be send to stdout? If NULL log is send to stdout.

Examples

```
data("ALL_dosages","integrated.maplist","LGHomDf_P1_1")
P1pp <- test_prefpairing(ALL_dosages,integrated.maplist,LGHomDf_P1_1,ploidy=4)
```

---

write.mct *Write MapChart file*

Description

Write a .mct file of a maplist for external plotting with MapChart software (Voorrips).

Usage

```
write.mct(maplist, mapdir = "mapping_files_MDSMap",
  file_info = paste("; MapChart file created on", Sys.Date()),
  filename = "MapFile", precision = 2, showMarkerNames = FALSE)
```

Arguments

maplist	A list of maps. In the first column marker names and in the second their position. All map data are compiled into a single MapChart file.
mapdir	Directory to which .mct files are written, by default the same directory as for MDSMap_from_list
file_info	A character string added to the first lines of the .mct file, by default a datestamp is recorded.
filename	Character string of filename to write the .mct file to, by default "MapFile"
precision	To how many decimal places should marker positions be specified (default = 2)?
showMarkerNames	Logical, by default FALSE, if TRUE, the marker names will be diplayed in the MapChart output as well.

Examples

```
data("integrated.maplist")
write.mct(integrated.maplist)
```

write.pwd *Write a JoinMap compatible .pwd file from linkage data.frame.*

Description

Output of this function allows to use JoinMap to perform the marker ordering step.

Usage

```
write.pwd(linkage_df, pwd_file, file_info, log = NULL)
```

Arguments

linkage_df	A linkage data.frame.
pwd_file	A character string specifying a file open for writing.
file_info	A character string added to the first lines of the .pwd file.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Examples

```
data("all_linkages_list_P1_split")
write.pwd(all_linkages_list_P1_split[["LG3"]][["homologue1"]],
          "LG3_homologue1_P1.pwd",
          "Please feed me to JoinMap")
```

write.TSNPM *Write TetraploidSNPMap input file*

Description

Output the phased linkage map files into format readable by TetraploidSNPMap (Hackett et al. 2017) to perform QTL analysis.

Usage

```
write.TSNPM(phased.maplist, outputdir = "TetraploidSNPMap_QTLfiles",
  filename = "TSNPM", ploidy, verbose = FALSE)
```

Arguments

phased.maplist	Phased maps in list format, the output of create_phased_maplist
outputdir	Directory to which TetraploidSNPMap files are written, by default written to "TetraploidSNPMap_QTLfiles" folder
filename	Character string of filename stem to write the output files to, by default "TSNPM" with linkage groups names appended
ploidy	The ploidy of the species, currently only 4 is supported by TetraploidSNPMap
verbose	Should messages be send to stdout?

Value

NULL

Examples

```
data("phased.maplist")
write.TSNPM(phased.maplist,ploidy=4)
```

---

write_nested_list *Write out a nested list*

Description

Write a nested list into a directory structure

Usage

```
write_nested_list(nested_list, directory, save_as_object = FALSE,
  object_prefix = directory, extension = if (save_as_object) ".Rdata" else
  ".txt", ...)
```

Arguments

nested_list	A nested list.
directory	Character string. Directory name to which to write the structure.
save_as_object	Logical. Save as R object?
object_prefix	Character. Prefix of R object. Only used if save_as_object = TRUE.
extension	Character. File extension. Default is ".txt".
...	Arguments passed to write.table

Examples

```
data("all_linkages_list_P1_subset")
write_nested_list(nested_list = all_linkages_list_P1_subset,
                  directory = "all_linkages_P1",
                  sep="\t")
```

write_pwd_list *Write pwd files from a nested list*

Description

A wrapper for write.pwd, which allows to write multiple pwd files with a directory structure according to the nested linkage list.

Usage

```
write_pwd_list(linkages_list, target_parent, binned = FALSE, dir = getwd(),
  log = NULL)
```

Arguments

linkages_list	A nested list with linkage group on the first level and homologue on the second.
target_parent	A character string specifying the name of the target parent.
binned	Logical. Are the markers binned? This information is used in the pwd header.
dir	A character string specifying the directory in which the files are written. Defaults to working directory.
log	Character string specifying the log filename to which standard output should be written. If NULL log is send to stdout.

Examples

```
data("all_linkages_list_P1_split")
write_pwd_list(all_linkages_list_P1_split, target_parent="P1", binned=FALSE)
```

Index