Package 'BatchMap'

November 12, 2017

Title Software for the Creation of High Density Linkage Maps in Outcrossing Species

Version 1.0.2.0

Description Algorithms that build on the 'OneMap' package to create linkage maps from high density data in outcrossing species in reasonable time frames.

Date 2017-10-30

LinkingTo Rcpp (>= 0.10.5), RcppArmadillo (>= 0.7.700)

Depends parallel (>= 3.2.3), ggplot2 (>= 1.0.1), R (>= 3.2.0)

Imports reshape2 (>= 1.4.1), methods

Suggests knitr (>= 1.10), rmarkdown

VignetteBuilder knitr

Encoding UTF-8

License GPL-3

URL https://github.com/bschiffthaler/BatchMap

BugReports https://github.com/bschiffthaler/BatchMap/issues

Maintainer Bastian Schiffthaler <bastian.schiffthaler@umu.se>

Repository CRAN

NeedsCompilation yes

Date/Publication 2017-11-11 23:55:01 UTC

RoxygenNote 6.0.1

Author Bastian Schiffthaler [aut, cre],
Gabriel Margarido [aut],
Marcelo Mollinari [aut],
Karl Broman [ctb],
Augusto Garcia [aut, ctb]

R topics documented:

`add.marker` *Creates a new sequence by adding markers.*

Description

Creates a new sequence by adding markers from a predetermined one. The markers are added in the end of the sequence.

Usage

```
add.marker(input.seq, mrks)
```

Arguments

`input.seq`	an object of class `sequence`.
`mrks`	a vector containing the markers to be added from the `sequence`.

Value

An object of class `sequence`, which is a list containing the following components:

`seq.num`	a `vector` containing the (ordered) indices of markers in the sequence, according to the input file.
`seq.phases`	a `vector` with the linkage phases between markers in the sequence, in corresponding positions. `-1` means that there are no defined linkage phases.
`seq.rf`	a `vector` with the recombination fractions between markers in the sequence. `-1` means that there are no estimated recombination fractions.
`seq.like`	log-likelihood of the corresponding linkage map.
`data.name`	name of the object of class `outcross` with the raw data.
`twopt`	name of the object of class `rf.2pts` with the 2-point analyses.

@author Marcelo Mollinari, <mmollina@usp.br>

See Also

`drop.marker`

Examples

```
data(example.out)
twopt <- rf.2pts(example.out)
all.mark <- make.seq(twopt,"all")
groups <- group(all.mark)
(LG1 <- make.seq(groups,1))
(LG.aug<-add.marker(LG1, c(4,7)))
```

`Bonferroni.alpha` *Calculates individual significance level to be used to achieve a global alpha (with Bonferroni)*

Description

It shows the alpha value to be used in each chi-square segregation test, in order to achieve a given global type I error. To do so, it uses Bonferroni's criteria.

Usage

```
Bonferroni.alpha(x, global.alpha = 0.05)
```

Arguments

`x`	an object of class onemap.segreg.test
`global.alpha`	the global alpha that

Value

the alpha value for each test (numeric)

`combine.onemap` *Combine OneMap datasets*

Description

Merge two or more OneMap datasets from the same cross type. Creates an object of class `onemap`.

Usage

```
combine.onemap(...)
```

Arguments

`...`	Two or more `onemap` dataset objects of the same cross type.

Details

Given a set of OneMap datasets, all from the same cross type (full-sib, backcross, F2 intercross or recombinant inbred lines obtained by self- or sib-mating), merges marker and phenotype information to create a single onemap object.

If sample IDs are present in all datasets (the standard new format), not all individuals need to be genotyped in all datasets - the merged dataset will contain all available information, with missing data elsewhere. If sample IDs are missing in at least one dataset, it is required that all datasets have the same number of individuals, and it is assumed that they are arranged in the same order in every dataset.

Value

An object of class onemap, i.e., a list with the following components:

geno	a matrix with integers indicating the genotypes read for each marker. Each column contains data for a marker and each row represents an individual.
n.ind	number of individuals.
n.mar	number of markers.
segr.type	a vector with the segregation type of each marker, as strings.
segr.type.num	a vector with the segregation type of each marker, represented in a simplified manner as integers, i.e. 1 corresponds to markers of type "A"; 2 corresponds to markers of type "B1.5"; 3 corresponds to markers of type "B2.6"; 4 corresponds to markers of type "B3.7"; 5 corresponds to markers of type "C.8"; 6 corresponds to markers of type "D1" and 7 corresponds to markers of type "D2". Markers for F2 intercrosses are coded as 1; all other crosses are left as NA.
input	a string indicating that this is a combined dataset.
n.phe	number of phenotypes.
pheno	a matrix with phenotypic values. Each column contains data for a trait and each row represents an individual.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

References

Lincoln, S. E., Daly, M. J. and Lander, E. S. (1993) Constructing genetic linkage maps with MAPMAKER/EXP Version 3.0: a tutorial and reference manual. *A Whitehead Institute for Biomedical Research Technical Report*.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

See Also

read.onemap and read.mapmaker.

Examples

```
  ## Not run:
    combined_data <- combine.onemap(onemap_data1, onemap_data2)

## End(Not run)
```

`compare` *Compare all possible orders (exhaustive search) for a given sequence of markers*

Description

For a given sequence with n markers, computes the multipoint likelihood of all $\frac{n!}{2}$ possible orders.

Usage

```
compare(input.seq, n.best = 50, tol = 0.001, verbose = FALSE)
```

Arguments

`input.seq`	an object of class `sequence`.
`n.best`	the number of best orders to store in object (defaults to 50).
`tol`	tolerance for the C routine, i.e., the value used to evaluate convergence.
`verbose`	if `FALSE` (default), simplified output is displayed. if `TRUE`, detailed output is displayed.

Details

Since the number $\frac{n!}{2}$ is large even for moderate values of n, this function is to be used only for sequences with relatively few markers. If markers were genotyped in an outcross population, linkage phases need to be estimated and therefore more states need to be visited in the Markov chain; when segregation types are D1, D2 and C, computation can required a very long time (specially when markers linked in repulsion are involved), so we recomend to use this function up to 6 or 7 markers. For inbred-based populations, up to 10 or 11 markers can be ordered with this function, since linkage phase are known. The multipoint likelihood is calculated according to Wu et al. (2002b) (Eqs. 7a to 11), assuming that the recombination fraction is the same in both parents. Hidden Markov chain codes adapted from Broman et al. (2008) were used.

Value

An object of class `compare`, which is a list containing the following components:

`best.ord`	a `matrix` containing the best orders.
`best.ord.rf`	a `matrix` with recombination frequencies for the corresponding best orders.

best.ord.phase	a matrix with linkage phases for the best orders.
best.ord.like	a vector with log-likelihood values for the best orders.
best.ord.LOD	a vector with LOD Score values for the best orders.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Jiang, C. and Zeng, Z.-B. (1997). Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. *Genetica* 101: 47-58.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. _Heredity_ 103: 494-502.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002a) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Wu, R., Ma, C.-X., Wu, S. S. and Zeng, Z.-B. (2002b). Linkage mapping of sex-specific differences. *Genetical Research* 79: 85-96

See Also

marker.type for details about segregation types and make.seq.

Examples

```
## Not run:
  #outcrossing example
  data(example.out)
  twopt <- rf.2pts(example.out)
  markers <- make.seq(twopt,c(12,14,15,26,28))
  (markers.comp <- compare(markers))
  (markers.comp <- compare(markers,verbose=TRUE))


## End(Not run)
```

`create.data.bins` *New dataset based on bins*

Description

Creates a new dataset based on `onemap.bin` object

Usage

```
create.data.bins(input.obj, bins)
```

Arguments

`input.obj`	an object of class `onemap`.
`bins`	an object of class `onemap.bin`.

Details

Given a `onemap.bin` object, creates a new data set where the redundant markers are collapsed into bins and represented by the marker with the lower amount of missing data among those on the bin.

Value

an object of class `onemap`.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

See Also

`find.bins`

Examples

```
 ## Not run:
  load(url("https://github.com/mmollina/data/raw/master/fake_big_data_f2.RData"))
  fake.big.data.f2
  (bins <- find.bins(fake.big.data.f2, exact=FALSE))
  (new.data <- create.data.bins(fake.big.data.f2, bins))
## End(Not run)
```

create.dataframe.for.plot.outcross *Create a dataframe suitable for a ggplot2 graphic*

Description

An internal function that prepares a dataframe suitable for drawing a graphic of raw data using ggplot2, i. e., a data frame with long format

Usage

```
create.dataframe.for.plot.outcross(x)
```

Arguments

x	an object of classes onemap and outcross, with data and additional information

Value

a dataframe

draw.map *Draw a genetic map*

Description

Provides a simple draw of a genetic map.

Usage

```
draw.map(map.list, horizontal = FALSE, names = FALSE, grid = FALSE,
  cex.mrk = 1, cex.grp = 0.75)
```

Arguments

map.list	a map, i.e. an object of class sequence with a predefined order, linkage phases, recombination fraction and likelihood; also it could be a list of maps.
horizontal	if TRUE, indicates that the map should be plotted horizontally. Default is FALSE
names	if TRUE, displays the names of the markers. Default is FALSE
grid	if TRUE, displays a grid in the background. Default is FALSE
cex.mrk	the magnification to be used for markers.
cex.grp	the magnification to be used for group axis annotation.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

Examples

```
## Not run:
 #outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  lg<-group(make.seq(twopt, "all"))
  maps<-vector("list", lg$n.groups)
  for(i in 1:lg$n.groups)
     maps[[i]]<- make.seq(order.seq(input.seq= make.seq(lg,i),twopt.alg =
   "rcd"), "force")
  draw.map(maps, grid=TRUE)
  draw.map(maps, grid=TRUE, horizontal=TRUE)


## End(Not run)
```

---

drop.marker *Creates a new sequence by dropping markers.*

Description

Creates a new sequence by dropping markers from a predetermined one.

Usage

```
drop.marker(input.seq, mrks)
```

Arguments

input.seq	an object of class sequence.
mrks	a vector containing the markers to be removed from the sequence.

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination fractions between markers in the sequence. -1 means that there are no estimated recombination fractions.

seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

@author Marcelo Mollinari, <mmollina@usp.br>

See Also

add.marker

Examples

```
data(example.out)
twopt <- rf.2pts(example.out)
all.mark <- make.seq(twopt,"all")
groups <- group(all.mark)
(LG1 <- make.seq(groups,1))
(LG.aug<-drop.marker(LG1, c(10,14)))
```

---

example.out *Data from a full-sib family derived from two outbred parents*

Description

Simulated data set for an outcross, i.e., an F1 population obtained by crossing two non-homozygous parents.

Usage

```
data(example.out)
```

Format

An object of class outcross.

Details

A total of 100 F1 individuals were genotyped for 30 markers. The data currently contains only genotype information (no phenotypes). It is included to be used as a reference in order to understand how a data file needs to be. Also, it is used for the analysis in the tutorial that comes with OneMap.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

See Also

read.outcross2 for details about objects of class outcross.

Examples

```
data(example.out)

# perform two-point analyses
twopts <- rf.2pts(example.out)
twopts
```

`find.bins` *Allocate markers into bins*

Description

Function to allocate markers with redundant information into bins. Within each bin, the pairwise recombination fraction between markers is zero.

Usage

```
find.bins(input.obj, exact = TRUE, ch = NULL)
```

Arguments

`input.obj`	an object of class `onemap`.
`exact`	logical. If `TRUE`, it only allocates markers with the exact same information into bins, including missing data; if `FALSE`, missing data are not considered when allocating markers. In the latter case, the marker with the lowest amount of missing data is taken as the representative marker on that bin.
`ch`	not used in this OneMap version. Chromosome for which the analysis should be performed. If `NULL` the analisys is performed for all chromosomes.

Value

An object of class `onemap.bin`, which is a list containing the following components:

`bins`	a list containing the bins. Each element of the list is a table whose lines indicate the name of the marker, the bin in which that particular marker was allocated and the percentage of missing data. The name of each element of the list corresponds to the marker with the lower amount of missing data among those on the bin
`n.mar`	total number of markers.
`n.ind`	number individuals
`exact.search`	logical; indicates if the search was performed with the argument `exact=TRUE` or `exact=FALSE`

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

See Also

create.data.bins

Examples

```
 ## Not run:
  load(url("https://github.com/mmollina/data/raw/master/fake_big_data_f2.RData"))
  fake.big.data.f2
  (bins<-find.bins(fake.big.data.f2, exact=FALSE))
## End(Not run)
```

---

group *Assign markers to linkage groups*

Description

Identifies linkage groups of markers, using results from two-point (pairwise) analysis and the *transitive* property of linkage.

Usage

```
group(input.seq, LOD = NULL, max.rf = NULL, verbose = TRUE)
```

Arguments

input.seq	an object of class sequence.
LOD	a (positive) real number used as minimum LOD score (threshold) to declare linkage.
max.rf	a real number (usually smaller than 0.5) used as maximum recombination fraction to declare linkage.
verbose	logical. If TRUE, current progress is shown; if FALSE, no output is produced.

Details

If the arguments specifying thresholds used to group markers, i.e., minimum LOD Score and maximum recombination fraction, are NULL (default), the values used are those contained in object input.seq. If not using NULL, the new values override the ones in object input.seq.

Value

Returns an object of class group, which is a list containing the following components:

data.name	name of the object of class onemap that contains the raw data.
twopt	name of the object of class rf.2ts used as input, i.e., containing information used to assign markers to linkage groups.
marnames	marker names, according to the input file.

n.mar	total number of markers.
LOD	minimum LOD Score to declare linkage.
max.rf	maximum recombination fraction to declare linkage.
n.groups	number of linkage groups found.
groups	number of the linkage group to which each marker is assigned.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com> and Marcelo Mollinari, <mmollina@usp.br>

References

Lincoln, S. E., Daly, M. J. and Lander, E. S. (1993) Constructing genetic linkage maps with MAPMAKER/EXP Version 3.0: a tutorial and reference manual. *A Whitehead Institute for Biomedical Research Technical Report*.

See Also

rf.2pts and make.seq

Examples

```
data(example.out)
twopts <- rf.2pts(example.out)

all.data <- make.seq(twopts,"all")
link_gr <- group(all.data)
link_gr
print(link_gr, details=FALSE) #omit the names of the markers
```

---

make.seq *Create a sequence of markers*

Description

Makes a sequence of markers based on an object of another type.

Usage

```
make.seq(input.obj, arg = NULL, phase = NULL, twopt = NULL)
```

Arguments

input.obj	an object of class rf.2pts, group, compare, try or order.
arg	its value depends on the type of object input.obj. For an object rf.2pts, arg can be the string "all", resulting in a sequence with all markers in the raw data (generally done for grouping markers); otherwise, it must be a vector of integers specifying which markers comprise the sequence. For an object of class group, arg must be an integer specifying the group. For a compare object, arg is an integer indicating the corresponding order (arranged according to the likelihood); if NULL (default), the best order is taken. For an object of class try, arg must be an integer less than or equal to the length of the original sequence plus one; the sequence obtained will be that with the additional marker in the position indicated by arg. Finally, for an order object, arg is a string: "safe" means the order that contains only markers mapped with the provided threshold; "force" means the order with all markers.
phase	its value is also dependent on the type of input.obj. For an rf.2pts object, phase can be a vector with user- defined linkage phases (its length is equal to the number of markers minus one); if NULL (default), other functions will try to find the best linkage phases. For example, if phase assumes the vector c(1,2,3,4), the sequence of linkage phases will be couple/couple, couple/repulsion, repulsion/couple and repulsion/repulsion for a sequence of five markers. If input.obj is of class compare or try, this argument indicates which combination of linkage phases should be chosen, for the particular order given by argument arg. In both cases, NULL (default) makes the best combination to be taken. If input.obj is of class group or order, this argument has no effect.
twopt	a string indicating the name of the object which contains the two-point information. This does not have to be defined by the user: it is here for compatibility issues when calling make.seq from inside other functions.

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.
seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Gabriel Margarido, <gramarga@gmail.com>

References

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

See Also

compare, try.seq, order.seq and map.

Examples

```
## Not run:
  data(example.out)
  twopt <- rf.2pts(example.out)

  all.mark <- make.seq(twopt,"all")
  all.mark <- make.seq(twopt,1:30) # same as above, for this data set
  groups <- group(all.mark)
  LG1 <- make.seq(groups,1)
  LG1.ord <- order.seq(LG1)
  (LG1.final <- make.seq(LG1.ord)) # safe order
  (LG1.final.all <- make.seq(LG1.ord,"force")) # forced order

  markers <- make.seq(twopt,c(2,3,12,14))
  markers.comp <- compare(markers)
  (base.map <- make.seq(markers.comp))
  base.map <- make.seq(markers.comp,1,1) # same as above
  (extend.map <- try.seq(base.map,30))
  (base.map <- make.seq(extend.map,5)) # fifth position is the best

## End(Not run)
```

---

map *Construct the linkage map for a sequence of markers*

Description

Estimates the multipoint log-likelihood, linkage phases and recombination frequencies for a sequence of markers in a given order.

Usage

```
map(input.seq, tol = 1e-04, verbosity = FALSE, phase.cores = 1)
```

Arguments

input.seq	an object of class sequence.
tol	tolerance for the C routine, i.e., the value used to evaluate convergence.
verbosity	Controls verbosity of phase. Currently can only be set to "phase"
phase.cores	Number of parallel cores used to estimate linkage phases. Should not be higher than 4.

Details

Markers are mapped in the order defined in the object input.seq. If this object also contains a user-defined combination of linkage phases, recombination frequencies and log-likelihood are estimated for that particular case. Otherwise, the best linkage phase combination is also estimated. The multipoint likelihood is calculated according to Wu et al. (2002b)(Eqs. 7a to 11), assuming that the recombination fraction is the same in both parents. Hidden Markov chain codes adapted from Broman et al. (2008) were used.

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.
seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Adapted from Karl Broman (package 'qtl') by Gabriel R A Margarido, <gramarga@usp.br> and Marcelo Mollinari, <mmollina@gmail.com>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Jiang, C. and Zeng, Z.-B. (1997). Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. *Genetica* 101: 47-58.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002a) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Wu, R., Ma, C.-X., Wu, S. S. and Zeng, Z.-B. (2002b). Linkage mapping of sex-specific differences. *Genetical Research* 79: 85-96

See Also

make.seq

Examples

```
data(example.out)
twopt <- rf.2pts(example.out)

markers <- make.seq(twopt,c(30,12,3,14,2)) # correct phases
map(markers)

markers <- make.seq(twopt,c(30,12,3,14,2),phase=c(4,1,4,3)) # incorrect phases
map(markers)
```

---

map.overlapping.batches

Mapping overlapping batches

Description

Apply the batch mapping algorithm using overlapping windows.

Usage

```
map.overlapping.batches(input.seq, size = 50, overlap = 15,
  fun.order = NULL, phase.cores = 1, ripple.cores = 1, verbosity = NULL,
  max.dist = Inf, ws = 4, increase.every = 4, max.tries = 10,
  min.tries = 0, seeds = NULL, optimize = "likelihood", ...)
```

Arguments

input.seq	an object of class sequence.
size	The center size around which an optimum is to be searched
overlap	The desired overlap between batches
fun.order	A function that is applied to each batch to improve marker order. See ripple.ord
phase.cores	The number of parallel processes to use when estimating the phase of a marker. (Should be no more than 4)

ripple.cores	The number of parallel processes to use when calculating alternative order.
verbosity	A character vector that includes any or all of "batch", "order", "position", "time" and "phase" to output progress status information.
max.dist	The maximum distance (in cM) two markers can have in a batch before automatic reordering is triggered (given that fun.order was set).
ws	The window size that the reordering function should use
increase.every	Increase the window size by one every n-th round when re-ordering.
max.tries	The maximum number of re-ordering tries. Failing to order after max.tries outputs a warning.
min.tries	The minimum number of re-ordering tries.
seeds	A vector of phase information used as seeds for the first batch
optimize	Either "likelihood" or "count". Passed to ripple.ord in order to optimize the map's likelihood or the RECORD COUNT criterion. Unless you are absolutely sure why, you should use "likelihood".
...	Other arguments passed to fun.order

Details

This algorithm implements the overlapping batch maps for high density marker sets. The mapping problem is reduced to a number of subsets (batches) which carry information forward in order to more accurately estimate recombination fractions and phasing. Further the user has the option of setting fun.order to a function that tries different orders and iteratively reorders markers to improve the map. See ripple.ord for such an implementation. The ordering function is triggered at least min.tries times per batch, or as long as a batch has two markers with a distance greater than max.dist.

Value

A list with the first element Map being an object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.
seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Secondly batches, a list of Maps for each of the batches.

Author(s)

Bastian Schiffthaler, <bastian.schiffthaler@umu.se>

See Also

pick.batch.sizes, map

---

map_func *Mapping functions Haldane and Kosambi*

Description

Functions to convert recombination fractions to distance in cM (centiMorgans).

Usage

```
haldane(rcmb)
kosambi(rcmb)
```

Arguments

rcmb	A recombination fraction between two markers, i.e., a number between 0 and 0.5.

Details

Haldane mapping function is defined as

$$d_M = -\frac{1}{2}\ln(1-2r),$$

for $0 \leq r \leq 0.5$, where r stands for the recombination fraction in rcmb. Kosambi mapping function is

$$d_M = \frac{1}{4}\ln\left[\frac{1+2r}{1-2r}\right],$$

for $0 \leq r \leq 0.5$, where r is defined as above.

Value

Both functions return a number with a distance measured in cM.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

References

Haldane, J. B. S. (1919) The combination of linkage values and the calculation of distance between the loci of linked factors. *Journal of Genetics* 8: 299-309.

Kosambi, D. D. (1944) The estimation of map distance from recombination values. *Annuaire of Eugenetics* 12: 172-175.

Examples

```
# little difference for small recombination fractions
haldane(0.05)
kosambi(0.05)

# greater difference as recombination fraction increases
haldane(0.35)
kosambi(0.35)
```

`marker.type` *Informs the segregation patterns of markers*

Description

Informs the type of segregation of all markers from an object of class `sequence`. For outcross populations it uses the notation by *Wu et al., 2002*. For backcrosses, F2s and RILs, it uses the traditional notation from MAPMAKER i.e. AA, AB, BB, not AA and not BB.

Usage

```
marker.type(input.seq)
```

Arguments

`input.seq`	an object of class `sequence`.

Details

The segregation types are (*Wu et al., 2002*):

Type	Cross	Segregation
A.1	ab x cd	1:1:1:1
A.2	ab x ac	1:1:1:1
A.3	ab x co	1:1:1:1
A.4	ao x bo	1:1:1:1
B1.5	ab x ao	1:2:1
B2.6	ao x ab	1:2:1
B3.7	ab x ab	1:2:1
C8	ao x ao	3:1
D1.9	ab x cc	1:1
D1.10	ab x aa	1:1
D1.11	ab x oo	1:1
D1.12	bo x aa	1:1
D1.13	ao x oo	1:1
D2.14	cc x ab	1:1
D2.15	aa x ab	1:1
D2.16	oo x ab	1:1
D2.17	aa x bo	1:1
D2.18	oo x ao	1:1

Value

Nothing is returned. Segregation types of all markers in the sequence are displayed on the screen.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

References

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

See Also

make.seq

Examples

```
data(example.out)
twopts <- rf.2pts(example.out)
markers.ex <- make.seq(twopts,c(3,6,8,12,16,25))
marker.type(markers.ex) # segregation type for some markers
```

---

order.seq *Search for the best order of markers combining compare and try.seq functions*

Description

For a given sequence of markers, this function first uses the compare function to create a framework for a subset of informative markers. Then, it tries to map remaining ones using the try.seq function.

Usage

```
order.seq(input.seq, n.init = 5, subset.search = c("twopt", "sample"),
  subset.n.try = 30, subset.THRES = 3, twopt.alg = c("rec", "rcd", "ser",
  "ug"), THRES = 3, touchdown = FALSE, wait = 0, tol = 0.1)
```

Arguments

input.seq	an object of class sequence.
n.init	the number of markers to be used in the compare step (defaults to 5).
subset.search	a character string indicating which method should be used to search for a subset of informative markers for the compare step. It is used for backcross, F_2 or RIL populations, but not for outcrosses. See the Details section.
subset.n.try	integer. The number of times to repeat the subset search procedure. It is only used if subset.search=="sample". See the Details section.
subset.THRES	numerical. The threshold for the subset search procedure. It is only used if subset.search=="sample". See the Details section.
twopt.alg	a character string indicating which two-point algorithm should be used if subset.search=="twopt". See the Details section.
THRES	threshold to be used when positioning markers in the try.seq step.
touchdown	logical. If FALSE (default), the try.seq step is run only once, with the value of THRES. If TRUE, try.seq runs with THRES and then once more, with THRES-1. The latter calculations take longer, but usually are able to map more markers. try.seq step is displayed. See Details section in try.seq function.
wait	the minimum time interval in seconds to display the diagnostic graphic for each try.seq step. Defaults to 0.00
tol	tolerance number for the C routine, i.e., the value used to evaluate convergence of the EM algorithm.

Details

For outcrossing populations, the initial subset and the order in which remaining markers will be used in the try.seq step is given by the degree of informativeness of markers (i.e markers of type A, B, C and D, in this order).

For backcrosses, F2s or RILs, two methods can be used for choosing the initial subset: i) "sample" randomly chooses a number of markers, indicated by n.init, and calculates the multipoint log-likelihood of the $\frac{n.init!}{2}$ possible orders. If the LOD Score of the second best order is greater than subset.THRES, than it takes the best order to proceed with the try.seq step. If not, the procedure is repeated. The maximum number of times to repeat this procedure is given by the subset.n.try argument. ii) "twopt" uses a two-point based algorithm, given by the option "twopt.alg", to construct a two-point based map. The options are "rec" for RECORD algorithm, "rcd" for Rapid Chain Delineation, "ser" for Seriation and "ug" for Unidirectional Growth. Then, equally spaced markers are taken from this map. The "compare" step will then be applied on this subset of markers.

In both cases, the order in which the other markers will be used in the try.seq step is given by marker types (i.e. co-dominant before dominant) and by the missing information on each marker.

After running the compare and try.seq steps, which result in a "safe" order, markers that could not be mapped are "forced" into the map, resulting in a map with all markers positioned.

Value

An object of class order, which is a list containing the following components:

ord	an object of class sequence containing the "safe" order.
mrk.unpos	a vector with unpositioned markers (if they exist).
LOD.unpos	a matrix with LOD-Scores for unmapped markers, if any, for each position in the "safe" order.
THRES	the same as the input value, just for printing.
ord.all	an object of class sequence containing the "forced" order, i.e., the best order with all markers.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Gabriel R A Margarido, <gramarga@usp.br> and Marcelo Mollinari, <mmollina@gmail.com>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Jiang, C. and Zeng, Z.-B. (1997). Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. *Genetica* 101: 47-58.

Lander, E. S. and Green, P. (1987). Construction of multilocus genetic linkage maps in humans. *Proc. Natl. Acad. Sci. USA* 84: 2363-2367.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002a) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Wu, R., Ma, C.-X., Wu, S. S. and Zeng, Z.-B. (2002b). Linkage mapping of sex-specific differences. *Genetical Research* 79: 85-96

See Also

make.seq, compare and try.seq.

Examples

```
## Not run:
  #outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  all.mark <- make.seq(twopt,"all")
  groups <- group(all.mark)
```

```
  LG2 <- make.seq(groups,2)
  LG2.ord <- order.seq(LG2,touchdown=TRUE)
  LG2.ord
  make.seq(LG2.ord) # get safe sequence
  make.seq(LG2.ord,"force") # get forced sequence


## End(Not run)
```

`pick.batch.sizes` *Picking optimal batch size values*

Description

Suggest an optimal batch size value for use in `map.overlapping.batches`

Usage

```
pick.batch.sizes(input.seq, size = 50, overlap = 15, around = 5)
```

Arguments

`input.seq`	an object of class `sequence`.
`size`	The center size around which an optimum is to be searched
`overlap`	The desired overlap between batches
`around`	The range around the center which is maximally allowed to be searched.

Value

An integer value for the size which most evenly divides batches. In case of ties, bigger batch sizes are preferred.

Author(s)

Bastian Schiffthaler, <bastian.schiffthaler@umu.se>

See Also

`map.overlapping.batches`

Examples

```
## Not run:
  LG <- structure(list(seq.num = seq(1,800)), class = "sequence")
  batchsize <- pick.batch.sizes(LG, 50, 19)

## End(Not run)
```

`plot.by.segreg.type` *Draw a graphic showing the number of markers of each segregation pattern.*

Description

The function receives an object of class onemap. For outcrossing populations, it can show detailed information (all 18 possible categories), or a simplified version.

Usage

```
## S3 method for class 'by.segreg.type'
plot(x, subcateg = TRUE, ...)
```

Arguments

x	an object of class onemap
subcateg	a TRUE/FALSE option to indicate if results will be plotted showing all possible categories (only for outcrossing populations)
...	Not used

Value

a ggplot graphic

Examples

```
data(example.out) #Outcrossing data
plot.by.segreg.type(example.out)
plot.by.segreg.type(example.out, subcateg=FALSE)

# You can store the graphic in an object, then save it.
# For details, see the help of ggplot2's function ggsave()
data(example.out) #Outcrossing data
g <- plot.by.segreg.type(example.out)
ggsave("SegregationTypes.jpg", g, width=7, height=4, dpi=600)
```

plot.onemap *Draw a graphic of raw data for any OneMap population*

Description

Shows a heatmap (in ggplot2, a graphic of geom "tile") for raw data. Lines correspond to markers and columns to individuals. The function can plot a graph for all marker types, depending of the cross type (dominant/codominant markers, in all combinations). The function receives a onemap object of class onemap, reads information from genotypes from this object, converts it to a long dataframe format using function melt() from package reshape2() or internal function create.dataframe.for.plot.outcross(), converts numbers from the object to genetic notation (according to the cross type), then plots the graphic. If there is more than 20 markers, removes y labels For outcross populations, it can show all markers together, or it can split them according the segregation pattern.

Usage

```
## S3 method for class 'onemap'
plot(x, all = TRUE, ...)
```

Arguments

x	an object of class onemap, with data and additional information
all	a TRUE/FALSE option to indicate if results will be plotted together (if TRUE) or splitted based on their segregation pattern. Only used for outcross populations.
...	currently ignored

Value

a ggplot graphic

Examples

```
data(example.out) # Loads a fake full-sib dataset installed with onemap
plot(example.out) # This will show you the graph for all markers
plot(example.out, all=FALSE) # This will show you the graph splitted for marker types

# You can store the graphic in an object, then save it.
# For details, see the help of ggplot2's function ggsave()
g <- plot(example.out, all=FALSE)
ggsave("MyRawData_out.jpg", g, width=9, height=4, dpi=600)
```

`plot.onemap.segreg.test` *Plot p-values for chi-square tests of expected segregation*

Description

Draw a graphic showing the p-values (re-scaled to -log10(p-values)) associated with the chi-square tests for the expected segregation patterns for all markers in a dataset. It includes a vertical line showing the threshold for declaring statistical significance if Bonferroni's correction is considered, as well as the percentage of markers that will be discarded if this criterion is used.

Usage

```
## S3 method for class 'onemap.segreg.test'
plot(x, order = TRUE, ...)
```

Arguments

`x`	an object of class onemap.segreg.test (produced by onemap's function test.segregation()), i. e., after performing segregation tests
`order`	a variable to define if p-values will be ordered in the plot
`...`	currently ignored

Value

a ggplot graphic

Examples

```
data(example.out) # load OneMap's fake dataset for an outcrossing population
Out.seg <- test.segregation(example.out) # Applies chi-square tests
print(Out.seg) # Shows the results
plot(Out.seg) # Plot the graph, ordering the p-values
plot(Out.seg, order=FALSE) # Plot the graph showing the results keeping the order in the dataset
# You can store the graphic in an object, then save it.
# For details, see the help of ggplot2's function ggsave()
g <- plot(Out.seg)
ggsave("SegregationTests.jpg", g, width=7, height=5, dpi=600)
```

print.group *Print group result*

Description

Generic print methods

Usage

```
## S3 method for class 'group'
print(x, detailed = TRUE, ...)
```

Arguments

x	The input object
detailed	Print detailed output
...	Not used

print.onemap.segreg.test *Print segregation test result*

Description

Generic print methods

Usage

```
## S3 method for class 'onemap.segreg.test'
print(x, ...)
```

Arguments

x	The input object
...	Not used

print.outcross *Print read.outcross result*

Description

Generic print methods

Usage

```
## S3 method for class 'outcross'
print(x, ...)
```

Arguments

x	The input object
...	Not used

print.rf.2pts *Print twopoint test result*

Description

Generic print methods

Usage

```
## S3 method for class 'rf.2pts'
print(x, mrk = NULL, ...)
```

Arguments

x	The input object
mrk	A marker position to print more detailed
...	Not used

print.try *Print try.seq result*

Description

Generic print methods

Usage

```
## S3 method for class 'try'
print(x, j = NULL, ...)
```

Arguments

x	The input object
j	A given position to print more detailed
...	Not used

pseudo.testcross.split *Split a dataset into parent-specific subsets*

Description

In order to create a map using the pseudo-testcross approach, the data needs to be split into parent specific subsets. This function uses the segregation type to create subset given an input object of type group.

Usage

```
pseudo.testcross.split(input.obj)
```

Arguments

input.obj	an object of class group

Value

A list with two marker sequences for each linkage group. One for sequences with type "D2.15", one for those with type "D1.10".

rcd *Rapid Chain Delineation*

Description

Implements the marker ordering algorithm *Rapid Chain Delineation* (*Doerge, 1996*).

Usage

```
rcd(input.seq, LOD = 0, max.rf = 0.5, tol = 1e-04)
```

Arguments

input.seq	an object of class sequence.
LOD	minimum LOD-Score threshold used when constructing the pairwise recombination fraction matrix.
max.rf	maximum recombination fraction threshold used as the LOD value above.
tol	tolerance for the C routine, i.e., the value used to evaluate convergence.

Details

Rapid Chain Delineation (*RCD*) is an algorithm for marker ordering in linkage groups. It is not an exhaustive search method and, therefore, is not computationally intensive. However, it does not guarantee that the best order is always found. The only requirement is a matrix with recombination fractions between markers. Next is an excerpt from QTL Cartographer Version 1.17 Manual describing the *RCD* algorithm (*Basten et al., 2005*):

The linkage group is initiated with the pair of markers having the smallest recombination fraction. The remaining markers are placed in a "pool" awaiting placement on the map. The linkage group is extended by adding markers from the pool of unlinked markers. Each terminal marker of the linkage group is a candidate for extension of the chain: The unlinked marker that has the smallest recombination fraction with either is added to the chain subject to the provision that the recombination fraction is statistically significant at a prespecified level. This process is repeated as long as markers can be added to the chain.

After determining the order with *RCD*, the final map is constructed using the multipoint approach (function map).

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.

seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

References

Basten, C. J., Weir, B. S. and Zeng, Z.-B. (2005) *QTL Cartographer Version 1.17: A Reference Manual and Tutorial for QTL Mapping*.

Doerge, R. W. (1996) Constructing genetic maps by rapid chain delineation. *Journal of Quantitative Trait Loci* 2: 121-132.

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

See Also

make.seq, map

Examples

```
## Not run:
  #outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  all.mark <- make.seq(twopt,"all")
  groups <- group(all.mark)
  LG1 <- make.seq(groups,1)
  LG1.rcd <- rcd(LG1)

## End(Not run)
```

---

read.mapmaker *Read data from a Mapmaker raw file*

Description

Imports data from a Mapmaker raw file.

Usage

```
read.mapmaker(dir, file)
```

Arguments

dir	directory where the input file is located.
file	the name of the input file which contains the data to be read.

Details

For details about MAPMAKER files see *Lincoln et al.* (1993). The current version supports backcross, F2s and RIL populations. The file can contain phenotypic data, but it will not be used in the analysis.

Value

An object of class onemap, i.e., a list with the following components:

geno	a matrix with integers indicating the genotypes read for each marker in onemap fashion. Each column contains data for a marker and each row represents an individual.
geno.mmk	a matrix with integers indicating the genotypes read for each marker in MAPMAKER/EXP fashion, i.e., 1, 2, 3: AA, AB, BB, respectively; 3, 4: BB, not BB, respectively; 1, 5: AA, not AA, respectively. Each column contains data for a marker and each row represents an individual.
n.ind	number of individuals.
n.mar	number of markers.
segr.type	a vector with the segregation type of each marker, as strings. Segregation types were adapted from outcross segregation types, using the same notation. For details see read.onemap.
segr.type.num	a vector with the segregation type of each marker, represented in a simplified manner as integers. Segregation types were adapted from outcross segregation types. For details see read.onemap.
input	the name of the input file.
n.phe	number of phenotypes.
pheno	a matrix with phenotypic values. Each column contains data for a trait and each row represents an individual. Currently ignored.

Author(s)

Adapted from Karl Broman (package **qtl**) by Marcelo Mollinari, <mmollina@usp.br>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Lincoln, S. E., Daly, M. J. and Lander, E. S. (1993) Constructing genetic linkage maps with MAPMAKER/EXP Version 3.0: a tutorial and reference manual. *A Whitehead Institute for Biomedical Research Technical Report*.

Examples

```
  ## Not run:
    map_data <-read.mapmaker(dir="work_directory",file="data_file.txt")

## End(Not run)
```

`read.onemap` *Read data from all types of progenies supported by OneMap*

Description

Imports data derived from outbred parents (full-sib family) or inbred parents (backcross, F2 intercross and recombinant inbred lines obtained by self- or sib-mating). Creates an object of class `onemap`.

Usage

```
read.onemap(dir, inputfile)
```

Arguments

`dir`	directory where the input file is located.
`inputfile`	the name of the input file which contains the data to be read.

Details

The file format is similar to that used by `MAPMAKER/EXP` (*Lincoln et al.*, 1993). The first line indicates the cross type, which must be one of `"outcross"`, `"f2 intercross"`, `"backcross"`, `"riself"` or `"risib"`. The second line contains five integers: i) the number of individuals; ii) the number of markers; iii) an indicator variable taking the value 1 if there is CHROM information, i.e., if markers are anchored on any reference sequence, and 0 otherwise; iv) a similar 1/0 variable indicating whether there is POS information for markers; and v) the number of phenotypic traits.

The next line contains sample IDs, separated by empty spaces or tabs. Addition of this sample ID requirement makes it possible for separate input datasets to be merged.

Next comes the genotype data for all markers. Each new marker is initiated with a "*" (without the quotes) followed by the marker name, without any space between them. Each marker name is followed by the corresponding segregation type, which may be: `"A.1"`, `"A.2"`, `"A.3"`, `"A.4"`, `"B1.5"`, `"B2.6"`, `"B3.7"`, `"C.8"`, `"D1.9"`, `"D1.10"`, `"D1.11"`, `"D1.12"`, `"D1.13"`, `"D2.14"`, `"D2.15"`, `"D2.16"`, `"D2.17"` or `"D2.18"` (without quotes), for full-sibs [see `marker.type` and *Wu et al.* (2002) for details]. Other cross types have special marker types: `"A.H"` for backcrosses; `"A.H.B"` for F2 intercrosses; and `"A.B"` for recombinant inbred lines.

After the segregation type comes the genotype data for the corresponding marker. Depending on the segregation type, genotypes may be denoted by `ac`, `ad`, `bc`, `bd`, `a`, `ba`, `b`, `bc`, `ab` and `o`, in several possible combinations. To make things easier, we have followed **exactly** the notation used by *Wu*

et al. (2002). Allowed values for backcrosses are a and ab; for F2 crosses they are a, ab and b; for RILs they may be a and b. Genotypes *must* be separated by a space. Missing values are denoted by "-".

If there is physical information for markers, i.e., if they are anchored at specific positions in reference sequences (usually chromosomes), this is included immediately after the marker data. These lines start with special keywords *CHROM and *POS and contain strings and integers, respectively, indicating the reference sequence and position for each marker. These also need to be separated by spaces.

Finally, if there is phenotypic data, it will be added just after the marker or CHROM/POS data. They need to be separated by spaces as well, using the same symbol for missing information.

The example directory in the package distribution contains an example data file to be read with this function. Further instructions can be found at the tutorial distributed along with this package.

Value

An object of class onemap, i.e., a list with the following components:

geno	a matrix with integers indicating the genotypes read for each marker. Each column contains data for a marker and each row represents an individual.
n.ind	number of individuals.
n.mar	number of markers.
segr.type	a vector with the segregation type of each marker, as strings.
segr.type.num	a vector with the segregation type of each marker, represented in a simplified manner as integers, i.e. 1 corresponds to markers of type "A"; 2 corresponds to markers of type "B1.5"; 3 corresponds to markers of type "B2.6"; 4 corresponds to markers of type "B3.7"; 5 corresponds to markers of type "C.8"; 6 corresponds to markers of type "D1" and 7 corresponds to markers of type "D2". Markers for F2 intercrosses are coded as 1; all other crosses are left as NA.
input	the name of the input file.
n.phe	number of phenotypes.
pheno	a matrix with phenotypic values. Each column contains data for a trait and each row represents an individual.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

References

Lincoln, S. E., Daly, M. J. and Lander, E. S. (1993) Constructing genetic linkage maps with MAPMAKER/EXP Version 3.0: a tutorial and reference manual. *A Whitehead Institute for Biomedical Research Technical Report*.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

See Also

combine.onemap and the example directory in the package source.

Examples

```
  ## Not run:
   outcr_data <- read.onemap(dir="work_directory", file="data_file.txt", cross="outcross")

## End(Not run)
```

---

read.outcross *Read data from a full-sib progeny (outcrossing populations)*

Description

Imports data from a full-sib family derived from the cross of two outbred parents and creates an object of class outcross.

Usage

```
read.outcross(dir, file)
```

Arguments

dir	directory where the input file is located.
file	the name of the input file which contains the data to be read.

Details

The file format is quite similar to that used by MAPMAKER/EXP (*Lincoln et al.*, 1993). The first line contains three integers: the number of individuals, the number of markers and the number of traits.

Next comes the genotype data for all markers. Each new marker is initiated with a "*" (without the quotes) followed by the marker name, without any space between them. Each marker name is followed by the corresponding segregation type, which may be: "A.1", "A.2", "A.3", "A.4", "B1.5", "B2.6", "B3.7", "C.8", "D1.9", "D1.10", "D1.11", "D1.12", "D1.13", "D2.14", "D2.15", "D2.16", "D2.17" or "D2.18" (without quotes) [see marker.type and *Wu et al.* (2002) for details].

After the segregation type comes the genotype data for the corresponding marker. Depending on the segregation type, genotypes may be denoted by ac, ad, bc, bd, a, ba, b, bc, ab and o, in several possible combinations. To make things easier, we have followed **exactly** the notation used by *Wu et al.* (2002). Genotypes *must* be separated by commas. Missing values are denoted by "-"

Finally, if there is phenotypic data, it will be added just after the marker data. They need to be separated by commas as well, using the same symbol for missing information.

The example directory in the package distribution contains an example data file to be read with this function. Further instructions can be found at the tutorial distributed along with the package.

Value

An object of class `outcross`, i.e., a list with the following components:

`geno`	a matrix with integers indicating the genotypes read for each marker. Each column contains data for a marker and each row represents an individual.
`n.ind`	number of individuals.
`n.mar`	number of markers.
`segr.type`	a vector with the segregation type of each marker, as `strings`.
`segr.type.num`	a vector with the segregation type of each marker, represented in a simplified manner as integers, i.e. 1 corresponds to markers of type `"A"`; 2 corresponds to markers of type `"B1.5"`; 3 corresponds to markers of type `"B2.6"`; 4 corresponds to markers of type `"B3.7"`; 5 corresponds to markers of type `"C.8"`; 6 corresponds to markers of type `"D1"` and 7 corresponds to markers of type `"D2"`
`n.phe`	the number of traits included in the file
`pheno`	the name of the phenoytpes
`input`	the name of the input file.

Author(s)

Adapted from Karl Broman (package **qtl**) by Gabriel R A Margarido, <gramarga@gmail.com>, later with additions from Luciano C Silva

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Lincoln, S. E., Daly, M. J. and Lander, E. S. (1993) Constructing genetic linkage maps with MAPMAKER/EXP Version 3.0: a tutorial and reference manual. *A Whitehead Institute for Biomedical Research Technical Report*.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

See Also

`example` directory in the package source.

Examples

```
  ## Not run:
    outcr_data <-
read.outcross(dir="work_directory",file="data_file.txt")

## End(Not run)
```

`read.outcross2` *Read data from a full-sib progeny (outcrossing populations)*

Description

This version implements the `read.outcross` function in a faster way. Everything else is essentially the same.

Usage

```
read.outcross2(infile)
```

Arguments

`infile`	the name of the input file which contains the data to be read.

Value

An object of class `outcross`, i.e., a list with the following components:

`geno`	a matrix with integers indicating the genotypes read for each marker. Each column contains data for a marker and each row represents an individual.
`n.ind`	number of individuals.
`n.mar`	number of markers.
`segr.type`	a vector with the segregation type of each marker, as `strings`.
`segr.type.num`	a vector with the segregation type of each marker, represented in a simplified manner as integers, i.e. 1 corresponds to markers of type `"A"`; 2 corresponds to markers of type `"B1.5"`; 3 corresponds to markers of type `"B2.6"`; 4 corresponds to markers of type `"B3.7"`; 5 corresponds to markers of type `"C.8"`; 6 corresponds to markers of type `"D1"` and 7 corresponds to markers of type `"D2"`
`n.phe`	the number of traits included in the file
`pheno`	the name of the phenoytpes
`input`	the name of the input file.

Author(s)

Adapted from Karl Broman (package **qtl**) by Gabriel R A Margarido, <gramarga@gmail.com>, later with additions from Luciano C Silva

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Lincoln, S. E., Daly, M. J. and Lander, E. S. (1993) Constructing genetic linkage maps with MAPMAKER/EXP Version 3.0: a tutorial and reference manual. *A Whitehead Institute for Biomedical Research Technical Report*.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

See Also

`example` directory in the package source.

Examples

```
## Not run:
    outcr_data <-
read.outcross2("data_file.txt")

## End(Not run)
```

`record` *Recombination Counting and Ordering*

Description

Implements the marker ordering algorithm *Recombination Counting and Ordering* (*Van Os et al., 2005*).

Usage

```
record(input.seq, times = 10, LOD = 0, max.rf = 0.5, tol = 1e-04)
```

Arguments

`input.seq`	an object of class `sequence`.
`times`	integer. Number of replicates of the RECORD procedure.
`LOD`	minimum LOD-Score threshold used when constructing the pairwise recombination fraction matrix.
`max.rf`	maximum recombination fraction threshold used as the LOD value above.
`tol`	tolerance for the C routine, i.e., the value used to evaluate convergence.

Details

Recombination Counting and Ordering (*RECORD*) is an algorithm for marker ordering in linkage groups. It is not an exhaustive search method and, therefore, is not computationally intensive. However, it does not guarantee that the best order is always found. The only requirement is a matrix with recombination fractions between markers.

After determining the order with *RECORD*, the final map is constructed using the multipoint approach (function `map`).

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.
seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

References

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

Van Os, H., Stam, P., Visser, R.G.F. and Van Eck, H.J. (2005) RECORD: a novel method for ordering loci on a genetic linkage map. *Theoretical and Applied Genetics* 112: 30-40.

See Also

make.seq and map

Examples

```
## Not run:
  ##outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  all.mark <- make.seq(twopt,"all")
  groups <- group(all.mark)
  LG1 <- make.seq(groups,1)
  LG1.rec <- record(LG1)

## End(Not run)
```

`record.parallel` *Recombination Counting and Ordering*

Description

Implements the marker ordering algorithm *Recombination Counting and Ordering* (*Van Os et al., 2005*).

Usage

```
record.parallel(input.seq, times = 10, cores = 1, LOD = 0, max.rf = 0.5,
  tol = 1e-04, useC = TRUE)
```

Arguments

`input.seq`	an object of class `sequence`.
`times`	integer. Number of replicates of the RECORD procedure.
`cores`	Number of parallel processes.
`LOD`	minimum LOD-Score threshold used when constructing the pairwise recombination fraction matrix.
`max.rf`	maximum recombination fraction threshold used as the LOD value above.
`tol`	tolerance for the C routine, i.e., the value used to evaluate convergence.
`useC`	Use the C implementation to get the matrix of recombination fractions.

Details

Recombination Counting and Ordering (*RECORD*) is an algorithm for marker ordering in linkage groups. It is not an exhaustive search method and, therefore, is not computationally intensive. However, it does not guarantee that the best order is always found. The only requirement is a matrix with recombination fractions between markers. This implementation parallelizes over the `times` argument using `cores` parallel processes. An optimal choice for the `cores` argument is usually an even divisor of `times`.

Value

An object of class `sequence`, which is a list containing the following components:

`seq.num`	a `vector` containing the (ordered) indices of markers in the sequence, according to the input file.
`seq.phases`	a `vector` with the linkage phases between markers in the sequence, in corresponding positions. `-1` means that there are no defined linkage phases.
`seq.rf`	a `vector` with the recombination frequencies between markers in the sequence. `-1` means that there are no estimated recombination frequencies.
`seq.like`	log-likelihood of the corresponding linkage map.
`data.name`	name of the object of class `outcross` with the raw data.
`twopt`	name of the object of class `rf.2pts` with the 2-point analyses.

Author(s)

Original implementation: Marcelo Mollinari, <mmollina@usp.br>, Parallel version: Bastian Schiffthaler, <bastian.schiffthaler@umu.se>

References

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

Van Os, H., Stam, P., Visser, R.G.F. and Van Eck, H.J. (2005) RECORD: a novel method for ordering loci on a genetic linkage map. *Theoretical and Applied Genetics* 112: 30-40.

See Also

`make.seq` and `map`

Examples

```
## Not run:
  ##outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  all.mark <- make.seq(twopt,"all")
  groups <- group(all.mark)
  LG1 <- make.seq(groups,1)
  LG1.rec <- record(LG1)

## End(Not run)
```

---

rf.2pts *Two-point analysis between genetic markers*

Description

Performs the two-point (pairwise) analysis proposed by *Wu et al. (2002)* between all pairs of markers.

Usage

```
rf.2pts(input.obj, LOD = 3, max.rf = 0.5, verbose = TRUE)
```

Arguments

`input.obj`	an object of class `onemap`.
`LOD`	minimum LOD Score to declare linkage (defaults to `3`).
`max.rf`	maximum recombination fraction to declare linkage (defaults to `0.50`).
`verbose`	logical. If `TRUE`, current progress is shown; if `FALSE`, no output is produced.

Details

For n markers, there are

$$\frac{n(n-1)}{2}$$

pairs of markers to be analyzed. Therefore, completion of the two-point analyses can take a long time.

Value

An object of class `rf.2pts`, which is a list containing the following components:

`n.mar`	total number of markers.
`LOD`	minimum LOD Score to declare linkage.
`max.rf`	maximum recombination fraction to declare linkage.
`input`	the name of the input file.
`analysis`	an array with the complete results of the two-point analysis for each pair of markers.

Note

The thresholds used for `LOD` and `max.rf` will be used in subsequent analyses, but can be overriden.

Author(s)

Gabriel R A Margarido <gramarga@gmail.com> and Marcelo Mollinari <mmollina@usp.br>

References

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Examples

```
data(example.out)

twopts <- rf.2pts(example.out,LOD=3,max.rf=0.5) # perform two-point analyses
twopts

print(twopts,c("M1","M2")) # detailed results for markers 1 and 2
```

ripple.ord *Update linkage map with alternative orders at all positions*

Description

This function carries out re-ordering of all markers using a sliding window according to user defined criteria. The best order is chosen based on the difference in log likelihood. Different heuristics are avaible to select which orders to test. Note that testing all orders has factorial complexity (N!/2) meaning that it's not feasible for window sizes larger than 6.

Usage

```
ripple.ord(input.seq, ws = 4, tol = 0.001, phase.cores = 1,
  ripple.cores = 1, method = "one", n = NULL, pref = "neutral",
  start = 1, verbosity = NULL, batches = NULL, no_reverse = TRUE,
  optimize = "likelihood")
```

Arguments

input.seq	An object of class sequence.
ws	The window size in which ti consider re-ordering
tol	The tolerance for checking convergence of the EM model
phase.cores	The number of parallel processes to use to estimate phases. Should be no higher than 4.
ripple.cores	The number of parallel processes that should be used when testing different marker orders.
method	One of "one", "all" or "rand". Which algorithm to use to generate alternative orders to test (see Details)
n	For method "rand": The number of random samples to be tested.
pref	For method "rand": One of "similar", "dissimilar" or "neutral". Controls if sampling probability should be adjusted based on similarity to the input sequence. See description.
start	The position of the first marker of the window within the input sequence
verbosity	A character vector that includes any or all of "batch", "order", "position", "time" and "phase" to output progress status information.
batches	List with batches that are being processed. Only used when estimating finish time.
no_reverse	For method "one": If FALSE, the method will also create all possible reverse sequences if the marker swaps.
optimize	Either "likelihood" or "count". Passed to ripple.ord in order to optimize the map's likelihood or the RECORD COUNT criterion. Unless you are absolutely sure why, you should use "likelihood".

Details

Methods: *all:*Checks for all possible permutations in the window. Will be very very slow for large window size. *one:*Checks for all possible pairwise switches in the window. The complexity scales as `sum(ws:1)`, `ws` being the window size. *rand:*First, generates all possible permutations. Then samples `n` sequences from those and tests those. Time complexity is N. The "rand" method can be further tuned to preferentially select similar, dissimilar sequences or to perform unbiased sampling. In the first two cases sampling probability is adjusted via a spearman correlation of the sequences to all possible sequences.

Value

An object of class `sequence` that is the best order for the re-ordered input sequence.

`ripple.seq` *Compares and displays plausible alternative orders for a given linkage group*

Description

For a given sequence of ordered markers, computes the multipoint likelihood of alternative orders, by shuffling subsets (windows) of markers within the sequence. For each position of the window, all possible $(ws)!$ orders are compared.

Usage

```
ripple.seq(input.seq, ws = 4, LOD = 3, tol = 0.1)
```

Arguments

`input.seq`	an object of class `sequence` with a predefined order.
`ws`	an integer specifying the length of the window size (defaults to 4).
`LOD`	threshold for the LOD-Score, so that alternative orders with LOD less then or equal to this threshold will be displayed.
`tol`	tolerance for the C routine, i.e., the value used to evaluate convergence.

Details

Large values for the window size make computations very slow, specially if there are many partially informative markers.

Value

This function does not return any value; it just produces text output to suggest alternative orders.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com> and Marcelo Mollinari, <mmollina@usp.br>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Jiang, C. and Zeng, Z.-B. (1997). Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. *Genetica* 101: 47-58.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002a) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Wu, R., Ma, C.-X., Wu, S. S. and Zeng, Z.-B. (2002b). Linkage mapping of sex-specific differences. *Genetical Research* 79: 85-96

See Also

`make.seq`, `compare`, `try.seq` and `order.seq`.

Examples

```
## Not run:
 #Outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  markers <- make.seq(twopt,c(27,16,20,4,19,21,23,9,24,29))
  markers.map <- map(markers)
  ripple.seq(markers.map)

## End(Not run)
```

`ripple.window` *Update linkage map with alternative orders at a given position*

Description

This function carries out re-ordering of one single window according to user defined criteria. The best order is chosen based on the difference in log likelihood. Different heuristics are avaible to select which orders to test. Note that testing all orders has factorial complexity (N!/2) meaning that it's not feasible for window sizes larger than 6.

Usage

```
ripple.window(input.seq, ws = 4, tol = 0.001, phase.cores = 1,
  ripple.cores = 1, start = 1, verbosity = NULL, type = "one",
  n = NULL, pref = NULL, no_reverse = TRUE, optimize = "likelihood")
```

Arguments

input.seq	An object of class sequence.
ws	The window size in which ti consider re-ordering
tol	The tolerance for checking convergence of the EM model
phase.cores	The number of parallel processes to use to estimate phases. Should be no higher than 4.
ripple.cores	The number of parallel processes that should be used when testing different marker orders.
start	The position of the first marker of the window within the input sequence
verbosity	A character vector that includes any or all of "batch", "order", "position", "time" and "phase" to output progress status information.
type	One of "one", "all" or "rand".
n	For method "rand": The number of random samples to be tested.
pref	For method "rand": One of "similar", "dissimilar" or "neutral". Controls if sampling probability should be adjusted based on similarity to the input sequence. See description.
no_reverse	For method "one": If FALSE, the method will also create all possible reverse sequences if the marker swaps.
optimize	Either "likelihood" or "count". Passed to ripple.ord in order to optimize the map's likelihood or the RECORD COUNT criterion. Unless you are absolutely sure why, you should use "likelihood".

Details

Methods: *all:*Checks for all possible permutations in the window. Will be very very slow for large window size. *one:*Checks for all possible pairwise switches in the window. The complexity scales as sum(ws:1), ws being the window size. *rand:*First, generates all possible permutations. Then samples n sequences from those and tests those. Time complexity is N.

The "rand" method can be further tuned to preferentially select similar, dissimilar sequences or to perform unbiased sampling. In the first two cases sampling probability is adjusted via a spearman correlation of the sequences to all possible sequences.

Value

An object of class sequence that is the best order for the re-ordered window within the input sequence.

`seeded.map` *Construct the linkage map for a sequence of markers after seeding phases*

Description

Estimates the multipoint log-likelihood, linkage phases and recombination frequencies for a sequence of markers in a given order using seeded phases.

Usage

```
seeded.map(input.seq, tol = 1e-04, phase.cores = 1, seeds,
  verbosity = NULL)
```

Arguments

`input.seq`	an object of class `sequence`.
`tol`	tolerance for the C routine, i.e., the value used to evaluate convergence.
`phase.cores`	The number of parallel processes to use when estimating the phase of a marker. (Should be no more than 4)
`seeds`	A vector given the integer encoding of phases for the first *N* positions of the map
`verbosity`	A character vector that includes any or all of "batch", "order", "position", "time" and "phase" to output progress status information.

Details

Markers are mapped in the order defined in the object `input.seq`. The best combination of linkage phases is also estimated starting from the first position not in the given seeds.The multipoint likelihood is calculated according to Wu et al. (2002b)(Eqs. 7a to 11), assuming that the recombination fraction is the same in both parents. Hidden Markov chain codes adapted from Broman et al. (2008) were used.

Value

An object of class `sequence`, which is a list containing the following components:

`seq.num`	a `vector` containing the (ordered) indices of markers in the sequence, according to the input file.
`seq.phases`	a `vector` with the linkage phases between markers in the sequence, in corresponding positions. `-1` means that there are no defined linkage phases.
`seq.rf`	a `vector` with the recombination frequencies between markers in the sequence. `-1` means that there are no estimated recombination frequencies.
`seq.like`	log-likelihood of the corresponding linkage map.
`data.name`	name of the object of class `outcross` with the raw data.
`twopt`	name of the object of class `rf.2pts` with the 2-point analyses.

Author(s)

Adapted from Karl Broman (package 'qtl') by Gabriel R A Margarido, <gramarga@usp.br> and Marcelo Mollinari, <mmollina@gmail.com>. Modified to use seeded phases by Bastian Schiffthaler <bastian.schiffthaler@umu.se>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Jiang, C. and Zeng, Z.-B. (1997). Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. *Genetica* 101: 47-58.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002a) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Wu, R., Ma, C.-X., Wu, S. S. and Zeng, Z.-B. (2002b). Linkage mapping of sex-specific differences. *Genetical Research* 79: 85-96

See Also

make.seq

Examples

```
data(example.out)
twopt <- rf.2pts(example.out)

markers <- make.seq(twopt,c(30,12,3,14,2))
seeded.map(markers, seeds = c(4,2))
```

---

select.segreg *Show markers with/without segregation distortion*

Description

A function to shows which marker have segregation distortion if Bonferroni's correction is applied for the Chi-square tests of mendelian segregation.

Usage

```
select.segreg(x, distorted = FALSE)
```

Arguments

x	an object of class onemap.segreg.test
distorted	a TRUE/FALSE variable to show distorted or non-distorted markers

Value

a vector with marker names, according to the option for "distorted"

---

seriation *Seriation*

Description

Implements the marker ordering algorithm *Seriation* (*Buetow & Chakravarti, 1987*).

Usage

```
seriation(input.seq, LOD = 0, max.rf = 0.5, tol = 1e-04)
```

Arguments

input.seq	an object of class sequence.
LOD	minimum LOD-Score threshold used when constructing the pairwise recombination fraction matrix.
max.rf	maximum recombination fraction threshold used as the LOD value above.
tol	tolerance for the C routine, i.e., the value used to evaluate convergence.

Details

Seriation is an algorithm for marker ordering in linkage groups. It is not an exhaustive search method and, therefore, is not computationally intensive. However, it does not guarantee that the best order is always found. The only requirement is a matrix with recombination fractions between markers.

NOTE: When there are to many pairs of markers with the same value in the recombination fraction matrix, it can result in ties during the ordination process and the *Seriation* algorithm may not work properly. This is particularly relevant for outcrossing populations with mixture of markers of type D1 and D2. When this occurs, the function shows the following error message: `There are too many ties in the ordination process - please, consider using another ordering algorithm`.

After determining the order with *Seriation*, the final map is constructed using the multipoint approach (function map).

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means that there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.
seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Gabriel R A Margarido, <gramarga@gmail.com>

References

Buetow, K. H. and Chakravarti, A. (1987) Multipoint gene mapping using seriation. I. General methods. *American Journal of Human Genetics* 41: 180-188.

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

See Also

make.seq, map

Examples

```
## Not run:
  ##outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  all.mark <- make.seq(twopt,"all")
  groups <- group(all.mark)
  LG3 <- make.seq(groups,3)
  LG3.ser <- seriation(LG3)

## End(Not run)
```

set.map.fun *Defines the default mapping function*

Description

Defines the function that should be used to display the genetic map through the analysis.

Usage

```
set.map.fun(type = c("kosambi", "haldane"))
```

Arguments

type	Indicates the function that should be used, which can be "kosambi" or "haldane"

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

References

Haldane, J. B. S. (1919) The combination of linkage values and the calculation of distance between the loci of linked factors. *Journal of Genetics* 8: 299-309.

Kosambi, D. D. (1944) The estimation of map distance from recombination values. *Annuaire of Eugenetics* 12: 172-175.

See Also

kosambi and haldane

suggest.lod *Suggests a LOD Score for two point tests*

Description

It suggests a LOD Score for declaring statistical significance for two-point tests for linkage between all pairs of markers, considering that multiple tests are being performed.

Usage

```
suggest.lod(x)
```

Arguments

x	an object of class onemap

Details

In a somehow naive approach, the function calculates the number of two-point tests that will be performed for all markers in the data set, and then using this to calculate the global alpha required to control type I error using Bonferroni's correction.

From this global alpha, the corresponding quantile from the chi-square distribution is taken and then converted to LOD Score.

This can be seen as just an initial approximation to help users to select a LOD Score for two point tests.

Value

the suggested LOD to be used for testing linkage

---

test.segregation *test.segregation*

Description

Using OneMap internal function test.segregation.of.a.marker(), performs the Chi-square test to check if all markers in a dataset are following the expected segregation pattern, i. e., 1:1:1:1 (A), 1:2:1 (B), 3:1 (C) and 1:1 (D) according to OneMap's notation.

Usage

```
test.segregation(x)
```

Arguments

x	an object of class onemap, with data and additional information.

Details

First, it identifies the correct segregation pattern and corresponding H0 hypothesis, and then tests it.

Value

an object of class onemap.segreg.test, which is a list with marker name, H0 hypothesis being tested, the chi-square statistics, the associated p-values and the % of individuals genotyped. To see the object, it is necessary to print it.

Examples

```
data(example.out) # Loads a fake outcross dataset installed with onemap
Chi <- test.segregation(example.out) # Performs the chi-square test for all markers
print(Chi) # Shows the results
```

`test.segregation.of.a.marker` *test.segregation.of.a.marker*

Description

Applies the chi-square test to check if markers are following the expected segregation pattern, i. e., 1:1:1:1 (A), 1:2:1 (B), 3:1 (C) and 1:1 (D) according to OneMap's notation. It does not use Yate's correction.

Usage

```
test.segregation.of.a.marker(x, marker)
```

Arguments

`x`	an object of class `onemap`, with data and additional information.
`marker`	the marker which will be tested for its segregation.

Details

First, the function selects the correct segregation pattern, then it defines the H0 hypothesis, and then tests it, together with percentage of missing data.

Value

a list with the H0 hypothesis being tested, the chi-square statistics, the associated p-values, and the % of individuals genotyped.

It returns `NA` if the numbers of expected and observed classes are different or if dominant and co-dominant coding is mixed in the same marker.

data(example.out) # Loads a fake outcross dataset installed with onemap test.segregation.of.a.marker(example.out,1)

`try.seq` *Try to map a marker into every possible position between markers in a given map*

Description

For a given linkage map, tries do add an additional unpositioned marker. This function estimates parameters for all possible maps including the new marker in all posible positions, while keeping the original linkage map unaltered.

Usage

```
try.seq(input.seq, mrk, tol = 0.1, pos = NULL, verbose = FALSE)
```

Arguments

input.seq	an object of class sequence with a predefined order.
mrk	the index of the marker to be tried, according to the input file.
tol	tolerance for the C routine, i.e., the value used to evaluate convergence.
pos	defines in which position the new marker mrk should be placed for the diagnostic graphic. If NULL (default), the marker is placed on the best position i.e. the one which results LOD = 0.00
verbose	if FALSE (default), simplified output is displayed. if TRUE, detailed output is displayed.

Details

The diagnostic graphic is made of three figures: i) the top figure represents the new genetic map obtained with the insertion of the new marker mrk on position pos. If pos = NULL (default), the marker is placed on the best position i.e. the one which results LOD = 0.00, which is indicated by a red triangle; ii) the left bottom figure represents the base map (contained in input.seq) on x-axis and the LOD-Scores of the linkage maps obtained with the new marker mrk tested at the beginning, between and at the end of the base map. Actually, it is a graphic representation of the LOD vector (see Value section). The red triangle indicates the best position where the new marker mrk should be placed; iii) the right bottom figure is the non-interactive rf.graph.table function for the new genetic map (deprecated in BatchMap). It plots a matrix of pairwise recombination fractions (under the diagonal) and LOD Scores (upper the diagonal) using a color scale.

Value

An object of class try, which is a list containing the following components:

ord	a list containing results for every linkage map estimated. These results include linkage phases, recombination frequencies and log-likelihoods.
LOD	a vector with LOD-Scores for each position where the additional marker is placed. This Score is based on the best combination of linkage phases for each map.
try.ord	a matrix with the orders of all linkage maps.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Jiang, C. and Zeng, Z.-B. (1997). Mapping quantitative trait loci with dominant and missing markers in various crosses from two inbred lines. *Genetica* 101: 47-58.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E. and Newburg, L. (1987) MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. *Genomics* 1: 174-181.

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetic maps. *Heredity* 103: 494-502

Wu, R., Ma, C.-X., Painter, I. and Zeng, Z.-B. (2002a) Simultaneous maximum likelihood estimation of linkage and linkage phases in outcrossing species. *Theoretical Population Biology* 61: 349-363.

Wu, R., Ma, C.-X., Wu, S. S. and Zeng, Z.-B. (2002b). Linkage mapping of sex-specific differences. *Genetical Research* 79: 85-96

See Also

make.seq and compare.

Examples

```
## Not run:
  #outcrossing example
  data(example.out)
  twopt <- rf.2pts(example.out)
  markers <- make.seq(twopt,c(2,3,12,14))
  markers.comp <- compare(markers)
  base.map <- make.seq(markers.comp,1)

  extend.map <- try.seq(base.map,30)
  extend.map
  print(extend.map,5) # best position
  print(extend.map,4) # second best position


## End(Not run)
```

---

ug *Unidirectional Growth*

Description

Implements the marker ordering algorithm *Unidirectional Growth* (*Tan & Fu, 2006*).

Usage

```
ug(input.seq, LOD = 0, max.rf = 0.5, tol = 1e-04)
```

Arguments

input.seq	an object of class sequence.
LOD	minimum LOD-Score threshold used when constructing the pairwise recombination fraction matrix.
max.rf	maximum recombination fraction threshold used as the LOD value above.
tol	tolerance for the C routine, i.e., the value used to evaluate convergence.

Details

Unidirectional Growth (*UG*) is an algorithm for marker ordering in linkage groups. It is not an exhaustive search method and, therefore, is not computationally intensive. However, it does not guarantee that the best order is always found. The only requirement is a matrix with recombination fractions between markers.

After determining the order with *UG*, the final map is constructed using the multipoint approach (function map).

Value

An object of class sequence, which is a list containing the following components:

seq.num	a vector containing the (ordered) indices of markers in the sequence, according to the input file.
seq.phases	a vector with the linkage phases between markers in the sequence, in corresponding positions. -1 means taht there are no defined linkage phases.
seq.rf	a vector with the recombination frequencies between markers in the sequence. -1 means that there are no estimated recombination frequencies.
seq.like	log-likelihood of the corresponding linkage map.
data.name	name of the object of class outcross with the raw data.
twopt	name of the object of class rf.2pts with the 2-point analyses.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

References

Mollinari, M., Margarido, G. R. A., Vencovsky, R. and Garcia, A. A. F. (2009) Evaluation of algorithms used to order markers on genetics maps. *Heredity* 103: 494-502.

Tan, Y. and Fu, Y. (2006) A novel method for estimating linkage maps. *Genetics* 173: 2383-2390.

See Also

make.seq, map

Examples

```
## Not run:
  #outcross example
  data(example.out)
  twopt <- rf.2pts(example.out)
  all.mark <- make.seq(twopt,"all")
  groups <- group(all.mark)
  LG1 <- make.seq(groups,1)
  LG1.ug <- ug(LG1)

## End(Not run)
```

`write.map` *Write a genetic map to a file*

Description

Write a genetic map to a file, base on a given map, or a list of maps. The output file can be used as an input to perform QTL mapping using the package R/qtl. It is also possible to create an output to be used with QTLCartographer program.

Usage

```
write.map(map.list, file.out)
```

Arguments

`map.list`	a map, i.e. an object of class `sequence` with a predefined order, linkage phases, recombination fraction and likelihood or a list of maps.
`file.out`	output map file.

Details

This function is avaliable only for backcross, F2 and RILs.

Author(s)

Marcelo Mollinari, <mmollina@usp.br>

References

Broman, K. W., Wu, H., Churchill, G., Sen, S., Yandell, B. (2008) *qtl: Tools for analyzing QTL experiments* R package version 1.09-43

Wang S., Basten, C. J. and Zeng Z.-B. (2010) Windows QTL Cartographer 2.5. Department of Statistics, North Carolina State University, Raleigh, NC.

Index