Package: polySegratio
Version: 0.2-1

January 9, 2008

R topics documented:

`addMisclass` *Misclassifies marker data in objects of class autoMarker or autoCross*

Description

Marker data are misclassified at a specified rate for objects of class `simAutoMarkers` or `simAutoCross`. The rate may be specified either as a proportion of missing at random or a proportion of columns and rows with specified proportions of missings.

Usage

```
addMisclass(x, misclass = 0, bands.missed=0, parents = FALSE,
parent.cols = c(1, 2), seed)
```

Arguments

x	object of class simAutoMarkers or simAutoCross, or a matrix with dominant markers scored as 0 or 1
misclass	proportion misclassified specified as for na.proportion (Default: 0)
bands.missed	proportion of bands that are not scored when they are actually present. Note this is applied to correctly specified markers after markers are misclassified (Default: 0)
parents	if TRUE then misclassify parental alleles, otherwise misclassify offspring marker alleles
parent.cols	for object of simAutoClass the columns containg parental markers
seed	random number generator (RNG) state for random number which will be set at start to reproduce results

Value

returns object of class simAutoMarkers or simAutoCross, or a matrix with dominant markers scored as 0 or 1 with extra components

misclass.info
: list with components

proportion	numeric proportion misclassified
index	indicates which markers were set as misclassified
bands.proportion	numeric proportion marker bands missed
bands.index	indicates which markers bands were missed
call	matches arguments when function called
time.generated	time/date when misclassifieds added
seed	seed for random number generation

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

addMissing add missing markers at random, sim.autoMarkers simulate autopolyploid markers, sim.autoCross simulate autopolyploid markers for a cross

Examples

```
## simulate autopolyploid markers
p1 <- sim.autoCross(4, dose.proportion=c(0.7,0.3), n.markers=20, n.indiv=10)
p2 <- sim.autoCross(4, dose.proportion=list(p01=c(0.7,0.3),p10=c(0.7,0.3),p11=c(
0.6,0.2,0.2)))

## add misclassified for a whopping 20% of markers
```

```
print(addMisclass(p1, 0.2, parents=TRUE), row=1:20)
addMisclass(p2, 0.1)
```

`addMissing` *Adds missing data to objects of class autoMarker or autoCross*

Description

Adds missing data to objects of class `simAutoMarkers` or `simAutoCross` as specified either as a proportion of missing at random or a proportion of columns and rows with specified proportions of missings.

Usage

```
addMissing(x, na.proportion = 0, parent.cols = c(1, 2), seed)
```

Arguments

`x`	object of class `simAutoMarkers` or `simAutoCross`, or a `matrix` with dominant markers scored as 0 or 1
`na.proportion`	proportion missing at random or a list with two components indiv and marker each containing c(prop. markers missing, prop. missing) (Default: 0)
`parent.cols`	columns containing parental markers (etc) not altered only used if object of class `simAutoCross`
`seed`	random number generator (RNG) state for random number which will be set at start to reproduce results

Value

Returns object of class `simAutoMarkers` or `simAutoCross`, or a `matrix` with dominant markers scored as 0 or 1 with extra component na.proportion which has the following elements

`na.proportion`	proportion missing at random or a list with two components indiv and marker each containing c(prop. markers missing, prop. missing)
`time.generated`	time/date when data set generated + when missing added
`seed`	random number generator seed which could be used to reproduce results (I hope)
`call`	matches arguments when function called

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`addMisclass` misclassifies markers at random, `sim.autoMarkers` simulate autopolyploid markers, `sim.autoCross` simulate autopolyploid markers for a cross

Examples

```
## simulate autopolyploid markers
p1 <- sim.autoCross(4, dose.proportion=c(0.7,0.3), n.markers=20, n.indiv=10)
p2 <- sim.autoCross(4, dose.proportion=list(p01=c(0.7,0.3),p10=c(0.7,0.3),p11=c(
0.6,0.2,0.2)))

## add missings
addMissing(p2, 0.1)
```

`autoFill` *Automatically fill out blanks of a vector with the preceding label*

Description

`autoFill` is commonly used to generate labels from columns of a spreadsheet when many cells are left blank in order to save a lot of typing. For instance, c("a","","","b","") becomes c("a","a","a","b","b")

Usage

```
autoFill(x, squash = FALSE)
```

Arguments

`x`	a vector of character strings
`squash`	If set to TRUE then leading and trailing spaces are removed which is useful if spaces are inadvertantly typed because these may be hard to track down. Default: FALSE

Value

`x`	a vector of character strings with blank strings replaced by preceding non–blank strings

Note

While this function may be called directly, it is more often called by `makeLabel`

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`makeLabel` uses `autoFill` to create labels from two columns of marker names

Examples

```
## description: fill out blanks of a vector with preceeding label
label.1 <- c("a","","","b","")
print(autoFill(label.1))

label.2 <- c("agc","","","","gct5","","ccc","","")
print(autoFill(label.2))
```

`divide.autoMarkers` *Divide markers by parental type*

Description

Given markers (or more correctly dominant 1,0) marker data and return list object of containing markers data split according to parental alleles, namely 1,0 for each parent and 1,1 for both parents

Usage

```
divide.autoMarkers(markers, description = paste("Markers split for",
deparse(substitute(markers))), parent.cols = c(1, 2),
extra.cols = NULL, cols.drop = c(parent.cols, extra.cols))
```

Arguments

`markers`	matrix of 1, 0, NA indicating marker alleles where rownames are markernames, column names are progeny names
`description`	text containing a description for printing
`parent.cols`	column(s) for parental markers (default: 1,2)
`extra.cols`	extra column(s) to be subsetted (default: NULL)
`cols.drop`	columns to be dropped from markers before splitting data which can be set to NULL if no columns are to be dropped (Default: c(parent.cols,extra.cols))

Value

Returns S3 class `divideAutoMarkers` containing

`p10, p01, p11`
: lists for where the first, second components are heterozygous for parents 1, 2 and both resp. Each list contains

`description`	text containing a description for printing
`parent`	label for parent
`markers`	markers for specified parental type (including parents etc)
`extras`	extra columns subsetted (if specified)
`seg.ratios`	segregation ratios as class `segRatio`

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`segRatio`, `sim.autoCross`

Examples

```
p2 <- sim.autoCross(4,
dose.proportion=list(p01=c(0.7,0.3),p10=c(0.7,0.3),
                     p11=c(0.6,0.2,0.2)))
print(p2)

ss <- divide.autoMarkers(p2$markers)

print(ss)
```

`divideAutoMarkers` *S3 class divideAutoMarkers*

Description

An S3 class which contains marker data and segregation proportions split into three groups corresponding to parents with '01', '10' and '11' markers

Value

`p10, p01, p11`
: lists for where the first, second components are heterozygous for parents 1, 2 and both resp. Each list contains

`description`
: text containing a description for printing

`parent`
: label for parent

`markers`
: markers for specified parental type (including parents etc)

`extras`
: extra columns subsetted (if specified)

`seg.ratios`
: segregation ratios as class `segRatio`

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`segRatio`, `sim.autoCross`

Examples

```
p2 <- sim.autoCross(4,
dose.proportion=list(p01=c(0.7,0.3),p10=c(0.7,0.3),
                     p11=c(0.6,0.2,0.2)))
print(p2)

ss <- divide.autoMarkers(p2$markers)

print(ss)
```

`expected.segRatio` *Compute theoretical segregation proportions for regular autopolyploids*

Description

Expected segregation proportions for various dosages of dominant markers for regular autopolyploids are calculated using the formula of Ripol et al (1999) based on Haldane (1930) for single dose and multiple dose parents cross nulliplex ("homozygous") and an unpublished formula where both parents possess at least single dose markers ("heterogenouous")

Usage

```
expected.segRatio(ploidy.level = stop("No ploidy level set"),
type.parents = c("heterogeneous", "homozygous"))
```

Arguments

`ploidy.level`	the number of homologous chromosomes, either as numeric or as a character string
`type.parents`	"heterogeneous" if parental markers are 0,1 or "homogeneous" if parental markers are both 1

Details

Value

`ratio`	vector of proportions for each dosage
`ploidy.level`	numeric value of ploidy level 2,4,6,8, ...
`ploidy.name`	name of ploidy

Warning

While results will be returned if the ploidy level is set as an odd number, the formula used are only for even numbers.

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

References

J B S Haldane (1930) Theoretical genetics of autopolyploids. *Journal of genetics* **22** 359–372

Ripol, M I et al (1999) Statistical aspects of genetic mapping in autopolyploids. *Gene* **235** 31–41

See Also

segRatio, test.segRatio

Examples

```
## heterogeneous parents

expected.segRatio(2)
expected.segRatio("Tetraploid")
expected.segRatio("tEtR")
expected.segRatio("octo")
expected.segRatio("Octa")
expected.segRatio(14)
## warning
expected.segRatio(9)

## errors - not run
## expected.segRatio("abcd")
## expected.segRatio(-1)

## homogeneous parents

expected.segRatio("Octa",type="heter")
expected.segRatio("Octa",type="homo")
expected.segRatio("tetra",type="homo")
expected.segRatio(6,type="homo")
expected.segRatio(9,type="homo")
```

---

makeLabel *Generate labels from two columns where blanks in first column are replaced by preceding non–blank label*

Description

Primarily used to generate marker labels from two columns where the first column is a nucleotide sequence which is mainly blank in that it is the same as the previous one while the second column is increasing numbers (fragment size) for each nucleotide combination

Usage

```
makeLabel(x, columns = c(1, 2), squash = TRUE, sep = "")
```

Arguments

x	data frame of markers including labels
columns	the column numbers containing labels (default: c(1,2))
squash	remove trailing/leading blanks in 1st column (default:TRUE)
sep	separator when combining two label columns (default: "")

Value

returns vector of marker names

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`autoFill` is used to replace blanks in first column

Examples

```
## imaginary data frame representing ceq marker names read in from
## spreadsheet
x <- data.frame( col1 = c("agc","","","","gct5","","ccc","",""),
                 col2 = c(1,3,4,5,1,2,2,4,6))
print(x)
print(makeLabel(x))
print(cbind(x,lab=makeLabel(x, sep=".")))
```

---

plot.segRatio *Plot segregation ratios for either observed or simulated marker data*

Description

Plots an object of S3 class `segRatio`

Usage

```
## S3 method for class 'segRatio':
plot(x, main =
deparse(substitute(x)), xlab="", xlab.segRatio = "Segregation ratio",
xlab.nobs = "Number of dominant markers",
xlab.miss = "Number of missing markers per individual",
NCLASS = 100, type = c("seg.ratio", "all","no","missing"), ...)

## S3 method for class 'simAutoMarkers':
plot(x, main = deparse(substitute(x)), xlab = "Segregation ratio",...)

## S3 method for class 'simAutoCross':
plot(x, main = deparse(substitute(x)), xlab = "Segregation ratio",
...)
```

Arguments

`x`	An object of class `segRatio`
`xlab`	label for x axis: not usually set
`main`	Title for plot
`xlab.segRatio`	x–axis label when plotting segregation proportions
`xlab.nobs`	x axis label when plotting no. of 1's
`xlab.miss`	x axis label when plotting number of missing individuals per marker

NCLASS	number of classes for histograms (Default: 100)
type	type of plot may be set to
seg.ratio	Histogram of segregation proportions (Default)
no	Histogram of the number of 1s
missing	Histogram of the numbers of missing values per marker
all	Produce all plots on one page
...	other parameters passed to plot function

Details

By default the histograms are produced of the segregation proportions. Other histograms that may be produced are numbers of observed dominant markers (recorded as a 1) and the number of individuals missing a particular marker.

Value

Used for its side-effects

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

segRatio, segregationRatios, sim.autoMarkers, sim.autoCross

Examples

```
## generate some autooctoploid data
a <- sim.autoMarkers(8,c(0.7,0.2,0.09,0.01))

## print markers and plot segratios
print(a)
plot(a$seg.ratios)  # plot the segregation ratios directly
plot(a)             # plot the simAutoMarkers object

## add some missing values and plot all histograms
plot(addMissing(a,0.2)$seg.ratios, type="all")
```

---

polySegratio-package

Segregation ratios for autoployploids

Description

These functions provide tools for computing expected segregation ratios (or more correctly segregation proportions) for dominant markers in regular autopolyploids and simulating such marker data as well as conducting standard Chi squared tests and Binomial confidance intervals for assigning marker dosage.

Details

Package:	polySegratio
Type:	Package
Version:	0.2-1
Date:	2008-01-08
License:	Copyright CSIRO and all rights reserved until further notice. Discussions needed.

Use `expected.segRatio` to compute expected segregation proportions for regular autopolyploids

Use `segregationRatios` to compute segregation ratios for a matrix of markers

Use `test.segRatio` to assignmarker dosage via Chi squared tests or Binomial CIs

Use `sim.autoMarkers` and `sim.autoCross` to simulate marker data under various scenarios

Use `addMisclass` and `addMissing` make some markers misclassified or missing at random

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

References

J B S Haldane (1930) Theoretical genetics of autopolyploids. *Journal of genetics* **22** 359–372

Ripol, M I et al (1999) Statistical aspects of genetic mapping in autopolyploids. *Gene* **235** 31–41

Examples

```
## expected segregation proportions heterogeneous parents
expected.segRatio(4)
expected.segRatio("Tetraploid")
expected.segRatio("Octa")

## expected segregation proportions homogeneous parents
expected.segRatio("Octa",type="heter")

## generate dominant markers for autotetraploids
a1 <- sim.autoMarkers(4,c(0.8,0.2))
print(a1)
plot(a1)

## generate crosses for different parental types
p2 <- sim.autoCross(4, dose.proportion=list(p01=c(0.7,0.3),
                    p10=c(0.7,0.3),p11=c(0.6,0.2,0.2)))
print(p2)
plot(p2)

## simulate and test some markers, printing out a summary table of
## no.s of correct marker dosages

a <- sim.autoMarkers(ploidy = 8, c(0.7,0.2,0.09,0.01),
                    type="hetero", n.markers=500,n.individuals=100)
a <- addMissing(a, 0.07)  # make seven percent missing at random
```

```
at <- test.segRatio(a$seg.ratios, ploidy=8, type.parents="het",
                        method="bin")
print(addmargins(table(a$true.doses$dosage, at$dosage, exclude=NULL)))
```

`print.segRatio` *Print segregation ratios*

Description

Prints an object of S3 class `segRatio`

Usage

```
## S3 method for class 'segRatio':
print(x, digits=3, ..., index = c(1:min(10,length(x$r))) )
```

Arguments

`x`	object of class `segRatio`
`digits`	minimal number of `significant` digits, see `print.default`
`index`	which rows of the marker matrix and segregation proportions to print. (Default: c(1:10))
`...`	extra parameters passed on to `print` function

Value

None.

Note

Objects of class `segRatio` may be produced from a `matrix` of markers by employing the function `segregationRatios`

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`segRatio`, `segregationRatios`, `print`,`print.default`

Examples

```
## generate autopolyploid markers
a1 <- sim.autoMarkers(4,c(0.8,0.2),n.markers=20,n.individuals=10)

print(class(a1$seg.ratios))
print(a1$seg.ratios)
```

print.simAutoMarkers *Print objects of class simAutoMarkers*

Description

Prints an object of S3 class simAutoMarkers

Usage

```
## S3 method for class 'simAutoMarkers':
print(x, ..., row.index = c(1:min(10,
nrow(x$markers))), col.index = c(1:min(10, ncol(x$markers))) )

## S3 method for class 'simAutoCross':
print(x, ..., row.index = c(1:min(10,
nrow(x$markers))), col.index = c(1:min(10, ncol(x$markers))))

## S3 method for class 'divideAutoMarkers':
print(x, ..., row.index = c(1:10),
  col.index = c(1:10), tabulate.extras = FALSE )
```

Arguments

x	object of class simAutoMarkers
row.index	which rows to print (Default: first 10)
col.index	which columns to print (Default: first 10)
tabulate.extras	If TRUE then cross–tabulate any axtra columns (Default: FALSE)
...	extra options for printing

Value

None.

Note

Objects of class simAutoMarkers may be produced from by employing the function sim.autoMarkers and the same for sim.autoCross and divide.autoMarkers

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

segRatio, segregationRatios, sim.autoCross, sim.autoMarkers, divide.autoMarkers, print

Examples

```
## generate data sets
a1 <- sim.autoMarkers(4,c(0.8,0.2))
a2 <- sim.autoMarkers(8,c(0.7,0.2,0.09,0.01),type="homo",n.markers=20,n.individuals=10)

print(a1)
print(a2)

## datasets from crosses
p1 <- sim.autoCross(4, dose.proportion=c(0.7,0.3), n.markers=20, n.indiv=10)
print(p1)
p2 <- sim.autoCross(4, dose.proportion=list(p01=c(0.7,0.3),p10=c(0.7,0.3),p11=c(0.6,0.2,0
print(p2)

## divide up data from crosses
ss <- divide.autoMarkers(p2$markers)
print(ss)
```

---

print.testSegRatio *Print objects of class testSegRatio*

Description

Prints an object of S3 class `testSegRatio`

Usage

```
## S3 method for class 'testSegRatio':
print(x, ..., last = 10)
```

Arguments

`x`	object of class `testSegRatio`
`last`	prints from 1 to `last` segregation ratio tests (Default: 10)
`...`	extra printing options

Value

None

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`segRatio`, `segregationRatios`, `test.segRatio`

Examples

```
## simulated data
a <- sim.autoMarkers(ploidy = 8, c(0.7,0.2,0.09,0.01))
ac <- test.segRatio(a$seg.ratios, ploidy=8, method="chi.squared")
print(ac)
```

---

segRatio *S3 class segRatio*

Description

An S3 class which contains the segregation ratios for dominant markers and other information such as the number of dominant markers per individual

Value

`r`	no. of 1's for each individual
`n`	total no. of markers present for each individual
`seg.ratio`	segregation proportion for each individual
`n.individuals`	total number of individuals

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`segregationRatios`: computing segregation ratios, `testSegRatio`: chi squared χ^2 and tests and Binomial confidence intervals for assigning marker dosage, `expected.segRatio`: compute expected segregation proportions for various dosages for dominant markers in regular autopolyploids

---

segregationRatios *Compute observed segregation proportions for dominant markers in autopolyploids*

Description

Computes segregation ratios for a matrix of markers where the rows are markers and the columns are individuals and the markers are recorded as 0's and 1's

Usage

```
segregationRatios(x, drop.cols = NULL)
```

Arguments

x	matrix of 0's, 1,s and NA's representing scores of dominant markers where the rows are markers and the columns are individuals
drop.cols	numeric columns to drop when calculating segregation ratios

Value

Returns an object of class segRatio containing

r	no. of 1's for each individual
n	total no. of markers present for each individual
seg.ratio	segregation proportion for each individual
n.individuals	total number of individuals

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

testSegRatio: chi squared χ^2 and tests and Binomial confidence intervals for assigning marker dosage, expected.segRatio: compute expected segregation proportions for various dosages for dominant markers in regular autopolyploids

Examples

```
## simulate small autotetraplid data set
a1 <- sim.autoMarkers(4,c(0.8,0.2),n.markers=20,n.individuals=10)
print(a1)

print(segregationRatios(a1$markers))
```

---

sim.autoCross *Simulate dominant markers for an autopolyploid cross for all parental types*

Description

Simulates dominant markers from an autopolyploid cross given the ploidy level and/or expected segregation ratios and the proportions in each dosage marker class. This is a wrapper to sim.autoMarkers to generate markers for '10', '01' and '11' parents

Usage

```
sim.autoCross(ploidy.level, prop.par.type = structure(c(0.4, 0.4, 0.2),
names = c("p10", "p01", "p11")), n.markers = 500, n.individuals = 200,
dose.proportion, true.seg.ratios, no.dosage.classes,
marker.names = paste("M", 1:n.markers, sep = "."),
individual.names = paste("X", 1:n.individuals, sep = "."),
parent.names = c("P.1", "P.2"), seed)
```

Arguments

`ploidy.level`	the number of homologous chromosomes, either as numeric (single value) or as a character string containing type tetraploid, hexaploid, octoploid, ...
`prop.par.type`	the proportion of markers generated from each parental type '10', '01' and '11'. Note that the exact number will be randomly generated from the multinomial distribution (Default: c(0.4,0.4,0.2))
`n.markers`	number of markers (Default: 500)
`n.individuals`	number of individuals in the cross (Default: 200)
`dose.proportion`	the proportion of markers to be simulated in each dosage class. Note that the exact number will be randomly generated from the multinomial distribution NB: If a vector is supplied the dose.proportion is same for each parental type otherwise as list with components 'p01', 'p10' and 'p11'
`true.seg.ratios`	numeric vector containing segregation proportion to be supplied if you wish to overide automatic calculations using ploidy.level
`no.dosage.classes`	numeric vector containing the number of dosage classes
`marker.names`	labels for markers (Default: M.1 ... M.n.markers)
`individual.names`	labels for offspring (Default: ... X.j ...)
`parent.names`	numeric vector of length 2 containing columns of marker matrix containing parental markers (Default: first 2 columns)
`seed`	integer used to set seed for random number generator (RNG) which (if set) may be used to reproduce results

Value

Returns an object of class `simAutoCross` containing

`markers`	matrix of 0,1 dominant markers with individuals as cols and rows as markers
`true.dosage`	*true* doses for each marker
`name.true.dose`	names of *true* doses for each marker
`p10`	object of class `simAutoMarkers` for parental type '10'
`p01`	object of class `simAutoMarkers` for parental type '01'
`p11`	object of class `simAutoMarkers` for parental type '11'
`ploidy.level`	the number of homologous chromosomes as numeric (single value)
`prop.par.type`	proportion of markers for each parental type 'p01', 'p10' and 'p11'
`n.markers`	number of markers (Default: 500)
`n.individuals`	number of individuals in the cross (Default: 200)
`dose.proportion`	proportion in each dose – if numeric vector is the same for 'p01', 'p10' and 'p11' else a list with components 'p01', 'p10' and 'p11'

`no.dosage.classes`
: number in each dosage class

`no.parType`
: number in each parental type

`time.generated`
: time/date when data set generated

`seed`
: seed for random number generator seed which could be used to reproduce results (I hope)

`call`
: matches arguments when function called

Note

All parameters except the proportions of marker dosage types can be left at the default. If only one value is set, then individual list components will be assumed to be equal. The marker matrix is prepended with parental marker alleles. An alternative is to simply create each group using sim.automarkers and cbind them.

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`simAutoCross`,`simAutoMarkers`, `sim.autoMarkers`

Examples

```
p1 <- sim.autoCross(4, dose.proportion=c(0.7,0.3), n.markers=20, n.indiv=10)
print(p1)

p2 <- sim.autoCross(4, dose.proportion=list(p01=c(0.7,0.3),p10=c(0.7,0.3),p11=c(0.6,0.2,0
print(p2)
```

`simAutoCross` *S3 class simAutoCross*

Description

An S3 class which contains simulated dominant marker data for autopolyploids and other data of interest such as segregation proportions as well as parameters set for the generating given parents with '01', '10' and '11' markers

Value

`markers`
: matrix of 0,1 dominant markers with individuals as cols and rows as markers

`true.dosage`
: *true* doses for each marker

`name.true.dose`
: names of *true* doses for each marker

`p10`
: object of class `simAutoMarkers` for parental type '10'

`p01`
: object of class `simAutoMarkers` for parental type '01'

`p11`
: object of class `simAutoMarkers` for parental type '11'

`ploidy.level` the number of homologous chromosomes as numeric (single value)

`prop.par.type`
: proportion of markers for each parental type 'p01', 'p10' and 'p11'

`n.markers` number of markers (Default: 500)

`n.individuals`
: number of individuals in the cross (Default: 200)

`dose.proportion`
: proportion in each dose – if numeric vector is the same for 'p01', 'p10' and 'p11' else a list with components sQuotep01, 'p10' and 'p11'

`no.dosage.classes`
: number in each dosage class

`no.parType` number in each parental type

`time.generated`
: time/date when data set generated

`seed` seed for random number generator seed which could be used to reproduce results (I hope)

`call` matches arguments when function called

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`sim.autoCross`,`simAutoMarkers`, `sim.autoMarkers`

`sim.autoMarkers` *Simulates dominant markers from an autopolyploid cross*

Description

Dominant markers are simulated from an autopolyploid cross given the ploidy level, expected segregation ratios and the proportions in each dosage marker class. This may be chosen from tetraploid to heccaidecaploid and the segregation ratios may be specified explicitly or generated automatically.

Usage

```
sim.autoMarkers(ploidy.level, dose.proportion, n.markers = 500,
n.individuals = 200, seg.ratios, no.dosage.classes,
type.parents = c("heterogeneous", "homozygous"),
marker.names = paste("M", 1:n.markers, sep = "."),
individual.names = paste("X", 1:n.individuals, sep = "."),
overdispersion=FALSE, shape1=50, seed)
```

Arguments

`ploidy.level`	the number of homologous chromosomes, either as numeric (single value) or as a character string containing type tetraploid, hexaploid, octoploid, ...
`dose.proportion`	the proportion of markers to be simulated in each dosage class. Note that the exact number will be randomly generated from the multinomial distribution
`n.markers`	number of markers (Default: 500)
`n.individuals`	number of individuals in the cross (Default: 200)
`seg.ratios`	numeric vector containing segregation proportion to be supplied if you wish to overide automatic calculations using ploidy.level
`no.dosage.classes`	only generate markers for the first `no.dosage.classes` classes (if set)
`type.parents`	heterogeneous for (1,0) or (0,1) homozygous for (1,1) (default: heterogeneous)
`marker.names`	labels for markers (Default: M.1 ... M.n.markers)
`individual.names`	labels for offspring (Default: ... X.j ...)
`overdispersion`	logical indicating overdispersion (Default: FALSE)
`shape1`	shape1 parameter(s) for the beta distribution used to generate the Binomial probability p, either of length 1 or no.dosage.classes. Default: 50 which implies very little overdispersion. NB: 'shape2' is calculated from shape 1 and expected segregation ratios
`seed`	integer used to set seed for random number generator (RNG) which (if set) may be used to reproduce results

Value

Returns an object of class `simAutoMarkers` containing

`markers`	matrix of 0,1 dominant markers with individuals as cols and rows as markers
`E.segRatio`	expected segregation porportions, list with components
`ratio`	segregation proportions,
`ploidy.level`	level of ploidy 4,6,8, ...
`ploidy.name`	tetraploid, ..., unknown
`type.parents`	heterogeneous for (1,0) or (0,1) homozygous for (1,1)
`dose.proportion`	proportions of markers set for each dosage class
`n.markers`	number of markers (Default: 500)
`n.individuals`	number of individuals in the cross (Default: 200)
`true.doses`	list containing
`dosage`	doses generated for each marker for simulation
`table.dosages`	summary of no.s in each dosage
`names`	names for each dosage such as SD (single dose), DD (double dose), SDxSD etc
`seg.ratios`	object of class segRatio containing segregation ratios

`time.generated`
: time/date when data set generated

`seed`
: seed for random number generator seed which could be used to reproduce results (I hope)

`overdispersion`
: either a list with components 'overdispersion': logical for whether overdispersion is set or not and if TRUE then two extra components 'shape1' and 'shape2' contain parameters for the beta distrubution employed to generate Binomial probabilities

`call`
: matches arguments when function called

Note

For use in simulation studies, other parameters such as the true dosage of each marker are also returned. Also, if extra binomial variation or overdispersion is requested then a beta-binomial distribution is employed to simulate marker data.Note that as the 'shape1' parameter becomes larger, the resulting marker data are less overdispersed.

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

`simAutoMarkers`, `print.simAutoMarkers`, `plot.simAutoMarkers`, `segRatio`

Examples

```
## generate autopolyploid markers
a1 <- sim.autoMarkers(4,c(0.8,0.2),n.markers=20,n.individuals=10)
print(a1)

a2 <-
sim.autoMarkers(8,c(0.7,0.2,0.09,0.01),type="homo",n.markers=20,n.individuals=10)
print(a2)
```

---

simAutoMarkers *S3 class simAutoMarkers*

Description

An S3 class which contains the simulated dominant marker data for autopolyploids and other data of interest such as segregation proportions as well as parameters set for the generating

Value

markers	matrix of 0,1 dominant markers with individuals as cols and rows as markers
E.segRatio	expected segregation porportions, list with components ratio: segregation proportions, ploidy.level: level of ploidy 4,6,8, ..., ploidy.name: tetraploid, ..., unknown
ploidy.level	the number of homologous chromosomes, either as numeric (single value) or as a character string containing type tetraploid, hexaploid, octoploid, ...
n.markers	number of markers (Default: 500)
n.individuals	number of individuals in the cross (Default: 200)
dose.proportion	the proportion of markers to be simulated in each dosage class. Note that the exact number will be randomly generated from the multinomial distribution
true.doses	list containing
dosage	doses generated for each marker for simulation
table.dosages	summary of no.s in each dosage
names	names for each dosage such as (SD) single dose, (DD) double dose, SDxSD etc
seg.ratios	segregation proportions as class segRatio
time.generated	date and time data set generated
call	function call used to generate data set

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

See Also

expected.segRatio, segRatio, print.simAutoMarkers, plot.simAutoMarkers

---

test.segRatio *Classic tests for assessing marker dosage in autopolyploids*

Description

Perform chi–squared tests or binomial CIs to obtain expected marker dosage in autopolyploids

Usage

```
test.segRatio(seg.ratio, ploidy.level = 4,
  type.parents = c("heterogeneous", "homozygous"),
  method = c("chi.squared", "binomial"), alpha = 0.05, expected.ratio)
```

Arguments

`seg.ratio`	object of class `segRatio` containing segregation proportions
`ploidy.level`	the number of homologous chromosomes, either as numeric or as a character string
`type.parents`	"heterogeneous" if parental markers are 0,1 or "homozygous" if parental markers are both 1
`method`	specify which method 'chi.squared' or 'binomial'
`alpha`	significance level for tests/CIs
`expected.ratio`	vector of expected segregation proportions Default: determined by using function `expected.segRatio` given the `ploidy.level`

Value

Returns object of class `testSegRatio` with components

`probability`	matrix of probabilities under the test for each dosage where columns are doses and rows are markers
`dosage`	vector of allocated dosages where allocation unique otherwise `NA`
`allocated`	matrix of 0's and 1's where 1 indicates dosage allocation where columns are doses and rows are markers
`alpha`	alpha level for significance test or CI construction
`expected.ratios`	expected segregation ratios under null hypotheses
`call`	call to test.segRatio

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

References

K Mather (1951) The measurement of linkage in heredity. *Methuen* London

Ripol, M I et al (1999) Statistical aspects of genetic mapping in autopolyploids. *Gene* **235** 31–41

See Also

`segregationRatios` for computing segregation ratios and `segRatio`, `expected.segRatio`

Examples

```
## simulated data
a <- sim.autoMarkers(ploidy = 8, c(0.7,0.2,0.09,0.01))
print(a)

## summarise chi-squared test vs true
ac <- test.segRatio(a$seg.ratios, ploidy=8, method="chi.squared")
print(addmargins(table(a$true.doses$dosage, ac$dosage, exclude=NULL)))
```

```
## summarise binomial CI vs true
ab <- test.segRatio(a$seg.ratios, ploidy=8, method="bin")
print(addmargins(table(a$true.doses$dosage, ab$dosage, exclude=NULL)))
```

`testSegRatio` *S3 class testSegRatio*

Description

An S3 class which contains results of classic tests for assessing marker dosage in autopolyploids using chi-squared tests or binomial confidence intervals

Value

Returns object of class `testSegRatio` with components

`probability`	matrix of probabilities under the test for each dosage where columns are doses and rows are markers
`dosage`	vector of allocated dosages where allocation unique otherwise `NA`
`allocated`	matrix of 0's and 1's where 1 indicates dosage allocation where columns are doses and rows are markers
`alpha`	alpha level for significance test or CI construction
`expected.ratios`	expected segregation ratios under null hypotheses
`call`	call to test.segRatio

Author(s)

Peter Baker ⟨Peter.Baker@csiro.au⟩

References

K Mather (1951) The measurement of linkage in heredity. *Methuen* London

Ripol, M I et al (1999) Statistical aspects of genetic mapping in autopolyploids. *Gene* **235** 31–41

See Also

`segRatio`, `expected.segRatio`, `test.segRatio`

Index