Package: simcross (via r-universe)

September 13, 2024

Version 0.6

Date 2023-11-29

Title Simulate Experimental Crosses

Description Simulate and plot general experimental crosses. The focus is on simulating genotypes with an aim towards flexibility rather than speed. Meiosis is simulated following the Stahl model, in which chiasma locations are the superposition of two processes: a proportion p coming from a process exhibiting no interference, and the remainder coming from a process following the chi-square model.

Author Karl W Broman [aut, cre] (<https://orcid.org/0000-0002-4914-6671>)

Maintainer Karl W Broman <broman@wisc.edu>

Depends R (>= 3.1.0)

Imports graphics, stats, Rcpp (>= 0.12.17)

Suggests qtl, knitr, rmarkdown, testthat, devtools, roxygen2

License GPL-3

URL https://kbroman.org/simcross/, https://github.com/kbroman/simcross

BugReports https://github.com/kbroman/simcross/issues

VignetteBuilder knitr

LinkingTo Rcpp

LazyData true

Encoding UTF-8

ByteCompile true

RoxygenNote 7.2.3

Roxygen list(markdown=TRUE)

Repository https://kbroman.r-universe.dev

RemoteUrl https://github.com/kbroman/simcross

RemoteRef HEAD

RemoteSha 1a4609495b392872cadb225f830e49809a2144ae

Contents

`AILped` *Example AIL pedigree*

Description

Example matrix describing the pedigree for advanced intercross lines

Usage

```
data(AILped)
```

Format

A data frame with five columns: individual id, mom, dad, sex (0 for females and 1 for males) and generation.

Source

Derived from the pedF8 dataset in the QTLRel package, https://cran.r-project.org/package=QTLRel

Examples

```
data(AILped)
x <- sim_from_pedigree(AILped)
```

`calc_Lstar` *Calculate adjusted chromosome length for obligate chiasma*

Description

Calculate the reduced chromosome length that will give the target expected number of chiasmata when conditioning on there being at least one chiasma on the four-strand bundle.

Usage

```
calc_Lstar(L, m = 0, p = 0)
```

Arguments

`L`	Length of chromosome (in cM); must be > 50
`m`	Interference parameter for chi-square model
`p`	Proportion of chiasmata coming from no-interference process

Value

Adjusted length of chromosome

See Also

`cross()`, `sim_meiosis()`, `sim_crossovers()`

Examples

```
calc_Lstar(100, 0, 0)
calc_Lstar(60, 10, 0.1)
```

CCcolors *Collaborative Cross colors*

Description

Get the vector of colors for the Collaborative Cross

Usage

```
CCcolors(palette = c("new", "original", "official"))
```

Arguments

palette	Which version of the colors to use? (New or original)

Value

vector of eight colors

Examples

```
CCcolors()
```

check_pedigree *Check a pedigree for errors*

Description

Perform a series of checks on the tabular data for a pedigree, checking for problems

Usage

```
check_pedigree(pedigree, ignore_sex = FALSE)
```

Arguments

pedigree	Numeric matrix or data frame with four columns: ID, mom ID, dad ID, sex. Sex is coded as 0=female, 1=male. There can be additional columns, but they'll be ignored.
ignore_sex	If TRUE, ignore the sex values completely (appropriate for hermaphroditic species.)

Details

The parents should be listed before any of their offspring. Founders should have 0's for mother and father; all others should have non-zero values for the parents, and the parents should appear in the pedigree. Father should be male and mothers should be female (unless ignore_sex=TRUE). Individual identifiers should be unique and non-zero. There should be no missing values anywhere. (NAs are allowed in the sex column if ignore_sex=TRUE.)

Value

TRUE (invisibly) if everything is okay; otherwise gives an error.

See Also

sim_from_pedigree(), sim_ril_pedigree()

Examples

```
tab <- sim_ril_pedigree(7)
check_pedigree(tab)
```

---

collapse_do_alleles *Collapse alleles for simulated DO genotypes*

Description

When simulating Diversity Outbreds, we need to specify parents 1-16, with 9-16 being the males from strains 1-8. This function collapses replaces alleles 9-16 with 1-8, to make the result ordinary DO-type data.

Usage

```
collapse_do_alleles(xodata)
```

Arguments

xodata	The sort of detailed genotype/crossover data generated by sim_from_pedigree().

Value

The input object, with alleles 9-16 replaced by 1-8.

See Also

sim_do_pedigree(), sim_do_pedigree_fix_n(), sim_from_pedigree()

Examples

```
# simulate DO pedigree
tab <- sim_do_pedigree(8)

# simulate genotypes for that pedigree
dat <- sim_from_pedigree(tab)
# collapse to alleles 1-8
dat <- collapse_do_alleles(dat)

# also works with data on multiple chromosomes
```

```
dat <- sim_from_pedigree(tab, c("1"=100, "2"=75, "X"=100), xchr="X")
dat <- collapse_do_alleles(dat)
```

---

convert2geno *Convert continuous allele information into marker genotypes*

Description

Convert the continuous crossover location information produced by sim_from_pedigree to marker genotypes

Usage

```
convert2geno(xodat, map, founder_geno = NULL, shift_map = FALSE)
```

Arguments

xodat	The sort of detailed genotype/crossover data generated by sim_from_pedigree()
map	vector of marker locations; can also be a list of such vectors (one per chromosome), in which case xodat and founder_geno must be lists with the same length.
founder_geno	Optional matrix (size n_founders x length(map)) of founder genotypes. If coded as 1/2 (or 1/3), results are 1/2/3 genotypes. If coded as A/T/G/C/N, results are A/T/G/C/N/H genotypes. If coded as letters A-H for the 8 founders, results are two-letter genotypes AA-HH with 36 possible values.
shift_map	If TRUE, shift genetic map to start at 0

Value

If founder_geno is provided or there are just two founders, the result is a numeric matrix of genotypes, individuals x markers, with genotypes 1/2/3 codes for 11/12/22 genotypes.

If founder_geno is not provided and there are more than two founders, the result is a 3-dimensional array, individuals x markers x alleles, with the third dimensional corresponding to the maternal and paternal allele.

If the input map is a list (the components being chromosomes), then xodat and founder_geno must be lists of the same length, and the result will be a list of matrices.

See Also

get_geno(), sim_from_pedigree()

Examples

```
# simulate AIL pedigree
tab <- sim_ail_pedigree(12, 30)
# simulate data from that pedigree
dat <- sim_from_pedigree(tab)
# marker map (could also use sim.map in R/qtl)
map <- seq(0, 100, by=5)
names(map) <- paste0("marker", seq(along=map))
# convert data to marker genotypes
geno <- convert2geno(dat, map)


# AIL with multiple chromosomes
dat <- sim_from_pedigree(tab, c("1"=100, "2"=75, "X"=100), xchr="X")
# marker map
multmap <- list("1"=seq(0, 100, by=5),
                "2"=seq(0, 75, by=5),
                "X"=seq(0, 100, by=5))
for(i in 1:3)
  names(multmap[[i]]) <- paste0("marker", i, "_", 1:length(map[[i]]))
geno <- convert2geno(dat, multmap)

# simulate DO pedigree
tab <- sim_do_pedigree(8)
# simulate data from that pedigree
dat <- sim_from_pedigree(tab)
# simulate founder snp alleles
fg <- matrix(sample(1:2, 8*length(map), repl=TRUE), nrow=8)
# for DO, need female & male founders (to deal with X chr)
fg <- rbind(fg, fg)
# convert dat to SNP genotypes
geno <- convert2geno(dat, map, fg)
# if fg not provided, result is a 3d array
genoarray <- convert2geno(dat, map)
```

`convert2geno_allchr` *Convert continuous allele information into marker genotypes for multiple chromosomes*

Description

Wrap up of convert2geno to adequate multiple chromosomes.

Usage

```
convert2geno_allchr(
  xodat,
  map,
  id = NULL,
```

```
  founder_geno = NULL,
  return.matrix = TRUE,
  shift_map = FALSE
)
```

Arguments

xodat	The sort of detailed genotype/crossover data generated by sim_from_pedigree_allchr()
map	marker locations, a list with elements for each chromosome
id	ids for which individuals genotypes is desired
founder_geno	Optional list of matrices (one per chromosome) of size n_founders x n_markers, with the founder genotypes. If coded as 1/2 (or 1/3), results are 1/2/3 genotypes. If coded as A/T/G/C/N, results are A/T/G/C/N/H genotypes. If coded as letters A-H (in the case of 8 founders), results are two-letter genotypes AA-HH with 36 possible values.
return.matrix	If FALSE, the result is a list of length n_chrs, otherwise it is converted into a matrix if size length(id) x n_markers.
shift_map	If TRUE, shift genetic map to start at 0

Value

If founder_geno is provided or there are just two founders, the result is a numeric matrix of genotypes, individuals x markers, with genotypes 1/2/3 codes for 11/12/22 genotypes. If there are more than two founders and founder_geno are letters, the result is a character matrix, too.

If founder_geno is not provided and there are more than two founders, the result is a 3-dimensional array, individuals x markers x alleles, with the third dimensional corresponding to the maternal and paternal allele.

See Also

convert2geno()

Examples

```
library(qtl)
# marker map
map <- sim.map(len=rep(100, 19), n.mar=10, include.x=FALSE)
# simulate AIL pedigree
tab <- sim_ail_pedigree(12, 30)
# simulate data from that pedigree
dat <- sim_from_pedigree_allchr(tab, map)
names(map) <- paste0("marker", seq(along=map))
# convert data to marker genotypes
id <- which(tab[, "gen"]==12)
geno <- convert2geno_allchr(dat, map, id)
```

create_parent *Create a parent object*

Description

Create a parent object

Usage

```
create_parent(L, allele = 1)
```

Arguments

L	chromosome length in cM
allele	vector of integers for alleles, of length 1 or 2

Value

A list with two components, for the individual's two chromosomes. Each is a list with alleles in chromosome intervals (as integers) and locations of the right endpoints of those intervals.

See Also

cross(), sim_meiosis()

Examples

```
create_parent(100, 1)
create_parent(100, 1:2)
```

cross *Cross two individuals*

Description

Simulate the cross of two individuals to create a single progeny

Usage

```
cross(
  mom,
  dad,
  m = 10,
  p = 0,
  xchr = FALSE,
  male = FALSE,
```

```
  obligate_chiasma = FALSE,
  Lstar = NULL
)
```

Arguments

mom	An individual object, as produced by create_parent() or this function.
dad	An individual object, as produced by create_parent() or this function.
m	interference parameter for chi-square model
p	proportion of crossovers coming from no-interference process
xchr	If TRUE, simulate X chromosome
male	If TRUE, simulate a male (matters only if xchr=TRUE)
obligate_chiasma	If TRUE, require an obligate chiasma on the 4-strand bundle at meiosis.
Lstar	Adjusted chromosome length, if obligate_chiasma=TRUE. Calculated if not provided.

Details

Simulations are under the Stahl model with the interference parameter being an integer. This is an extension of the chi-square model, but with chiasmata being the superposition of two processes, one following the chi-square model and the other exhibiting no interference.

Value

A list with two components, for the individual's two chromosomes. Each is a list with alleles in chromosome intervals (as integers) and locations of the right endpoints of those intervals.

See Also

create_parent(), sim_meiosis(), sim_crossovers(), calc_Lstar()

Examples

```
mom <- create_parent(100, 1:2)
dad <- create_parent(100, 1:2)
child <- cross(mom, dad)
```

get_geno *Get genotype at a single position*

Description

With data on the continuous crossover location information produced by sim_from_pedigree, grab the genotype at a given position.

Usage

```
get_geno(xodat, position)
```

Arguments

xodat	The sort of detailed genotype/XO data generated by sim_from_pedigree()
position	Position (in cM) for which to obtain genotypes

Value

A numeric matrix with two columns: the maternal and paternal allele for each individual.

See Also

sim_from_pedigree(), convert2geno()

Examples

```
# simulate AIL pedigree
tab <- sim_ail_pedigree(12, 30)
# simulate data from that pedigree
dat <- sim_from_pedigree(tab)
# get genotype at position 30 cM
geno <- get_geno(dat, 30)
```

mouseL_cox *Mouse chromosome lengths*

Description

Mouse chromosome lengths in cM from the Cox et al. map

Usage

```
data(mouseL_cox)
```

Format

A numeric vector with lengths in cM for the 20 mouse chromosomes.

Source

Taken from Table 1 of Cox et al. (2009) A new standard genetic map for the laboratory mouse. Genetics 182:1335-1344. doi:10.1534/genetics.109.105486

See Also

mouseL_mgi

Examples

```
data(mouseL_cox)
```

---

mouseL_mgi *Mouse chromosome lengths*

Description

Mouse chromosome lengths in cM from the Mouse Genome Informatics (MGI) standard map.

Usage

```
data(mouseL_mgi)
```

Format

A numeric vector with lengths in cM for the 20 mouse chromosomes.

Source

Taken from Table 1 of Cox et al. (2009) A new standard genetic map for the laboratory mouse. Genetics 182:1335-1344. doi:10.1534/genetics.109.105486

See Also

mouseL_cox

Examples

```
data(mouseL_mgi)
```

plot_crosslines *Plot cross lines*

Description

Add lines for a cross

Usage

```
plot_crosslines(
  momloc,
  dadloc,
  kidsloc,
  gap = 3,
  chrlength = 30,
  cex = 1.5,
  lwd = 2,
  arrow_length = 0.1,
  col = "white",
  ...
)
```

Arguments

momloc	An (x,y) vector with center location for mother
dadloc	An (x,y) vector with center location for mother
kidsloc	Either an (x,y) vector with center location for a kid, or a list of such for multiple kids
gap	Gap arrows and points/rectangles
chrlength	Length of chromosomes
cex	Character expansion for x point
lwd	Line width for points, segments, and arrows
arrow_length	The length parameter in the call to graphics::arrows()
col	Color of lines and points
...	Additional arguments passed to arrows() and segments()

Value

None.

See Also

plot_ind()

Examples

```
mom <- create_parent(100, 1:2)
dad <- create_parent(100, 3:4)
kids <- lapply(1:4, function(junk) cross(mom, dad))
plot(0,0, type="n", xlim=c(0, 100), ylim=c(0,100),
     xaxt="n", yaxt="n", xlab="", ylab="")
loc <- list(c(25,75), c(75,75), c(12.5,25), c(37.5,25), c(62.5, 25), c(87.5,25))
plot_ind(mom, loc[[1]])
plot_ind(dad, loc[[2]])
for(i in 1:4) plot_ind(kids[[i]], loc[[i+2]])
plot_crosslines(loc[[1]], loc[[2]], loc[3:6])
```

`plot_ind` *Plot an individual*

Description

Add an individual, as a pair of chromosomes, to a plot

Usage

```
plot_ind(
  ind,
  center,
  chrlength = 30,
  chrwidth = 3,
  gap = 3,
  col = CCcolors(),
  border = "black",
  lend = 1,
  ljoin = 1,
  allborders = FALSE,
  ...
)
```

Arguments

`ind`	An individual object, as output by `create_parent()` or `cross()`
`center`	(x,y) vector for the center of the individual
`chrlength`	Length of chromosomes (Can be a vector of length 2, in which case the two chromosomes will be different lengths, aligned at the top. This is for the X chromosome.)
`chrwidth`	Width of chromosomes
`gap`	Gap between chromosomes
`col`	Vector of colors
`border`	Color for border

lend	Passed to graphics::rect()
ljoin	Passed to graphics::rect()
allborders	If TRUE, put borders around all segments
...	Additional arguments passed to rect()

Value

None.

See Also

plot_crosslines()

Examples

```
mom <- create_parent(100, 1:2)
dad <- create_parent(100, 3:4)
kid <- cross(mom, dad)
plot(0,0, type="n", xlim=c(0, 100), ylim=c(0,100),
     xaxt="n", yaxt="n", xlab="", ylab="")
loc <- list(c(25,75), c(75,75), c(50,25))
plot_ind(mom, loc[[1]])
plot_ind(dad, loc[[2]])
plot_ind(kid, loc[[3]])
plot_crosslines(loc[[1]], loc[[2]], loc[[3]])
```

---

sim_4way_pedigree *Simulate pedigree for 4-way intercross*

Description

Simulate a 4-way cross, among four inbred lines (a table of individual, mom, dad, sex)

Usage

```
sim_4way_pedigree(ngen = 1, nsibs = 100)
```

Arguments

ngen	Number of intercross generations (1 or 2)
nsibs	Vector with number of siblings in the sibships in the last generation.

Details

We start with a set of 4 individuals (representing four inbred lines), and make a pair of crosses to generate a pair of heterozygous individuals. These are then crosses to generate a set of F1 individuals. If ngen==1, we stop there, with sum(nsibs) individuals in this last generation. If gen==2, we generate length(nsibs) male/female pairs of F1 offspring; these are intercrossed to generate a set of sibships, with lengths defined by the values in nsibs. Individuals in the last generation are alternating female/male.

Value

A data frame with five columns: individual ID, mother ID, father ID, sex, and generation. Founders have `0` for mother and father ID. Sex is coded 0 for female and 1 for male.

See Also

`sim_from_pedigree()`, `sim_ril_pedigree()`, `sim_do_pedigree()`, `sim_ail_pedigree()`

Examples

```
# 100 F1s between heterozygous parents
tab <- sim_4way_pedigree(1, 100)
# could also do this
tab2 <- sim_4way_pedigree(1, rep(10, 10))

# 120 F2s in 10 sibships each of size 12
tab3 <- sim_4way_pedigree(ngen=2, rep(12, 10))
```

---

sim_ail_pedigree *Simulate AIL pedigree*

Description

Simulate a pedigree for advanced intercross lines (a table of individual, mom, dad, sex)

Usage

```
sim_ail_pedigree(
  ngen = 12,
  npairs = 30,
  nkids_per = 5,
  design = c("nosib", "random")
)
```

Arguments

`ngen`	Number of generations of outbreeding
`npairs`	Number of breeding pairs at each generation
`nkids_per`	Number of offspring per pair for the last generation
`design`	How to choose crosses: either random but avoiding siblings, or completely at random

Details

Advanced intercross lines (AIL) are generated from a pair of inbred lines. We cross them and then cross the F1 to generate `npair` breeding pairs. The subsequent `ngen` outbreeding generations then proceed by crossing a male and female from the preceding generation (mated completely at random, with `design="random"`, or avoiding siblings, with `design="nosib"`). Each breeding pair gives a single female and a single male to the next generation, except at the last generation `nkids_per` offspring are mated, in equal numbers male and female. (If `nkids_per` is an odd number, the number of males and females in each sibship will differ by one, alternating between sibships, with one additional female and then one additional male.

Value

A data frame with five columns: individual ID, mother ID, father ID, sex, and generation. Founders have `0` for mother and father ID. Sex is coded 0 for female and 1 for male.

See Also

`sim_from_pedigree()`, `sim_ril_pedigree()`, `sim_do_pedigree()`, `sim_4way_pedigree()`

Examples

```
tab <- sim_ail_pedigree(12, 30)
```

---

sim_ail_pedigree_fix_n

Simulate AIL pedigree with fixed n

Description

Simulate a pedigree for advanced intercross lines (a table of individual, mom, dad, sex) so that the last generation reaches a desired sample size n

Usage

```
sim_ail_pedigree_fix_n(
  ngen = 12,
  nkids_per = 5,
  nsample_ngen = 150,
  npairs = NULL,
  method = c("last2", "sub2"),
  design = c("nosib", "random")
)
```

Arguments

ngen	Number of generations of outbreeding
nkids_per	Number of offspring per pair for the last generation
nsample_ngen	Number of individuals desired at the last generation
npairs	Number of breeding pairs at each generation. If missing, we use 30 when method="last2" and 300 when method="sub2".
method	Method used to generate pedigree: either expand at the last two generations or generate a pedigree with a large number of pairs and select a subset to have the desired sample size.
design	How to choose crosses: either random but avoiding siblings, or completely at random

Details

The default value for npairs depends on the choice of method. For method="last2", we use a default of npairs=30; for method="sub2", we use a default of npairs=300.

Value

A data frame with five columns: individual ID, mother ID, father ID, sex, and generation. Founders have 0 for mother and father ID. Sex is coded 0 for female and 1 for male.

See Also

sim_from_pedigree(), sim_ril_pedigree(), sim_ail_pedigree(), sim_do_pedigree(), sim_4way_pedigree(), sim_do_pedigree_fix_n()

Examples

```
tab <- sim_ail_pedigree_fix_n(12)
```

---

sim_crossovers *Simulate crossover locations using the Stahl model*

Description

Simulate crossover locations on a single meiotic product using the Stahl model.

Usage

```
sim_crossovers(L, m = 10, p = 0, obligate_chiasma = FALSE, Lstar = NULL)
```

Arguments

L	length of chr in cM
m	Interference parameter (m=0 is no interference)
p	Proportion of chiasmata from no-interference mechanism (p=0 gives pure chi-square model)
obligate_chiasma	If TRUE, require an obligate chiasma on the 4-strand bundle at meiosis.
Lstar	Adjusted chromosome length, if obligate_chiasma=TRUE. Calculated if not provided.

Details

Chiasma locations are a superposition of two processes: a proportion p exhibiting no interference, and a proportion (1-p) following the chi-square model with interference parameter m. Crossover locations are derived by thinning the chiasma locations with probability 1/2.

Simulations are under the Stahl model with the interference parameter being an integer. This is an extension of the chi-square model, but with chiasmata being the superposition of two processes, one following the chi-square model and the other exhibiting no interference.

Value

Numeric vector of crossover locations, in cM

References

Copenhaver, G. P., Housworth, E. A. and Stahl, F. W. (2002) Crossover interference in arabidopsis. *Genetics* **160**, 1631–1639.

Foss, E., Lande, R., Stahl, F. W. and Steinberg, C. M. (1993) Chiasma interference as a function of genetic distance. *Genetics* **133**, 681–691.

Zhao, H., Speed, T. P. and McPeek, M. S. (1995) Statistical analysis of crossover interference using the chi-square model. *Genetics* **139**, 1045–1056.

Examples

```
x <- sim_crossovers(200, 10, 0)
x <- sim_crossovers(200, 10, 0.04)
x <- sim_crossovers(100, 0, 0, obligate_chiasma=TRUE)
```

`sim_dof1_pedigree` *Simulate pedigree for F1 between diversity outbreds and another inbred line*

Description

Simulate a pedigree for a set of DOF1 individuals: the F1 offspring of a set of diversity outbred mice and another inbred strain (such as a mutant line).

Usage

```
sim_dof1_pedigree(
  ngen = 12,
  npairs = 144,
  ccgen = rep(4:12, c(21, 64, 24, 10, 5, 9, 5, 3, 3)),
  nkids_per = 5,
  design = c("nosib", "random")
)
```

Arguments

`ngen`	Number of generations of outbreeding
`npairs`	Number of breeding pairs at each generation
`ccgen`	Vector of length `npairs`, with the number of generations for each CC line. If length 1, it is repeated to a vector of length `npairs`.
`nkids_per`	Number of offspring per pair for the last DO generation (each will be crossed to produce one F1)
`design`	How to choose crosses: either random but avoiding siblings, or completely at random

Details

Diversity outbred (DO) mice are generated from a set of 8 inbred lines. We need two individuals from each line (one female and one male) as the order of the initial crosses will be randomized; for example, sometimes the individual from line 1 will be a mother and sometimes a father. The founders are numbered 1-8 for the females from the 8 lines, and 9-16 for the corresponding males.

Diversity Outbred mice are generated by first creating a panel of partially-inbred 8-way RIL (the so-called pre-CC, for pre-Collaborative Cross). The `ccgen` argument specifies the number of inbreeding generations for each of the CC lines. We generate a pre-CC line for each of the `npairs` breeding pairs, and generate a sibling pair from each as the starting material.

The subsequent `ngen` outbreeding generations then proceed by crossing a male and female from the preceding generation (mated completely at random, with `design="random"`, or avoiding siblings, with `design="nosib"`). Each breeding pair gives a single female and a single male to the next generation, except at the last generation `nkids_per` offspring are mated, in equal numbers male and female. (If `nkids_per` is an odd number, the number of males and females in each sibship

will differ by one, alternating between sibships, with one additional female and then one additional male.

The default for ccgen is taken from Figure 1 of Svenson et al. (2012).

We assume that the F1 offspring are all from a cross DO female x line 17 male, and so the last generation of the DO is taken to be all females.

Value

A data frame with seven columns: individual ID, mother ID, father ID, sex, generation, a TRUE/FALSE indicator for whether DO or pre-DO, and a TRUE/FALSE indicator for whether DOF1. Founders have `0` for mother and father ID. Sex is coded 0 for female and 1 for male.

References

Svenson KL, Gatti DM, Valdar W, Welsh CE, Cheng R, Chesler EJ, Palmer AA, McMillan L, Churchill GA (2012) High-resolution genetic mapping using the mouse Diversity Outbred population. Genetics 190:437-447

See Also

sim_from_pedigree(), sim_ril_pedigree(), sim_ail_pedigree(), sim_4way_pedigree()

Examples

```
tab <- sim_dof1_pedigree(8)
```

---

sim_do_pedigree *Simulate a pedigree for Diversity Outbred mice*

Description

Simulate a pedigree for generating Diversity Outbred (DO) mice (a table of individual, mom, dad, sex).

Usage

```
sim_do_pedigree(
  ngen = 12,
  npairs = 144,
  ccgen = rep(4:12, c(21, 64, 24, 10, 5, 9, 5, 3, 3)),
  nkids_per = 5,
  design = c("nosib", "random")
)
```

Arguments

ngen	Number of generations of outbreeding
npairs	Number of breeding pairs at each generation
ccgen	Vector of length npairs, with the number of generations for each CC line. If length 1, it is repeated to a vector of length npairs.
nkids_per	Number of offspring per pair for the last generation
design	How to choose crosses: either random but avoiding siblings, or completely at random

Details

Diversity outbred (DO) mice are generated from a set of 8 inbred lines. We need two individuals from each line (one female and one male) as the order of the initial crosses will be randomized; for example, sometimes the individual from line 1 will be a mother and sometimes a father. The founders are numbered 1-8 for the females from the 8 lines, and 9-16 for the corresponding males.

Diversity Outbred mice are generated by first creating a panel of partially-inbred 8-way RIL (the so-called pre-CC, for pre-Collaborative Cross). The ccgen argument specifies the number of inbreeding generations for each of the CC lines. We generate a pre-CC line for each of the npairs breeding pairs, and generate a sibling pair from each as the starting material.

The subsequent ngen outbreeding generations then proceed by crossing a male and female from the preceding generation (mated completely at random, with design="random", or avoiding siblings, with design="nosib"). Each breeding pair gives a single female and a single male to the next generation, except at the last generation nkids_per offspring are mated, in equal numbers male and female. (If nkids_per is an odd number, the number of males and females in each sibship will differ by one, alternating between sibships, with one additional female and then one additional male.

The default for ccgen is taken from Figure 1 of Svenson et al. (2012).

Value

A data frame with six columns: individual ID, mother ID, father ID, sex, generation, and TRUE/FALSE indicator for whether DO or pre-DO. Founders have 0 for mother and father ID. Sex is coded 0 for female and 1 for male.

References

Svenson KL, Gatti DM, Valdar W, Welsh CE, Cheng R, Chesler EJ, Palmer AA, McMillan L, Churchill GA (2012) High-resolution genetic mapping using the mouse Diversity Outbred population. Genetics 190:437-447

See Also

sim_from_pedigree(), sim_ril_pedigree(), sim_ail_pedigree(), sim_4way_pedigree()

Examples

```
tab <- sim_do_pedigree(8)
```

`sim_do_pedigree_fix_n` *Simulate a pedigree for Diversity Outbreds for a target sample size*

Description

Simulate a pedigree for Diversity Outbred (DO) mice (a table of individual, mom, dad, sex) so that the last generation reaches a desired sample size.

Usage

```
sim_do_pedigree_fix_n(
  ngen = 12,
  nkids_per = 5,
  nccgen = 15,
  nsample_ngen = 150,
  npairs = NULL,
  method = c("last2", "sub2", "fixcc"),
  design = c("nosib", "random"),
  selc.method = c("byfamily", "byindiv")
)
```

Arguments

`ngen`	Number of generations of outbreeding
`nkids_per`	Number of offspring per pair for the last generation
`nccgen`	The number of generations for each CC line, only used when `method` is not `"fixcc"`.
`nsample_ngen`	Number of individuals desired at the last generation
`npairs`	Number of breeding pairs at each generation. If missing, we use 30 when `method="last2"` and 300 when `method="sub2"`.
`method`	Method used to generate the pedigree: either expand at the last two generations or generate a pedigree with a large number of pairs and then select a subset to have the desired sample size. With `method="fixcc"`, we use the pre-CC generations as performed at the Jackson Lab.
`design`	How to choose crosses: either random but avoiding siblings, or completely at random
`selc.method`	Method used to select the individuals from last generation.

Details

The default number of breeding pairs depends on the chosen `method`. With `method="last2"`, the default is `npairs=30`; with `method="sub2"`, the default is `npairs=300`; with `method="fixcc"`, `npairs` is ignored and is fixed at 144.

Value

A data frame with six columns: individual ID, mother ID, father ID, sex, generation, and TRUE/FALSE indicator for whether DO or pre-DO. Founders have `0` for mother and father ID. Sex is coded 0 for female and 1 for male.

See Also

`sim_from_pedigree()`, `sim_ril_pedigree()`, `sim_ail_pedigree()`, `sim_do_pedigree()`, `sim_4way_pedigree()`, `sim_ail_pedigree_fix_n()`

Examples

```
tab <- sim_do_pedigree_fix_n(8)
```

---

sim_from_pedigree *Simulate genotypes for pedigree*

Description

Simulate genotypes along one chromosome for a pedigree

Usage

```
sim_from_pedigree(
  pedigree,
  L = 100,
  xchr = FALSE,
  m = 10,
  p = 0,
  obligate_chiasma = FALSE
)
```

Arguments

`pedigree`	Matrix or data frame describing a pedigree, with first four columns being individual ID, mom ID, dad ID, and sex (female as `0`, male as `1`).
`L`	Length of chromosome in cM (or a vector of chromosome lengths)
`xchr`	If TRUE, simulate X chromosome. (If `L` is a vector, this should be a vector of TRUE/FALSE values, of the same length as `L`, or a character string with the name of the X chromosome, in `L`.)
`m`	Crossover interference parameter, for chi-square model (m=0 corresponds to no interference).
`p`	proportion of crossovers coming from no-interference process
`obligate_chiasma`	If TRUE, require an obligate chiasma on the 4-strand bundle at meiosis.

Value

A list with each component being the data for one individual, as produced by the `cross()` function. Those results are a list with two components, corresponding to the maternal and paternal chromosomes. The chromosomes are represented as lists with two components: an integer vector of alleles in chromosome intervals, and a numeric vector of locations of the right-endpoints of those intervals; these two vectors should have the same length.

If the input `L` is a vector, in order to simulate multiple chromosomes at once, then the output will be a list with length `length(L)`, each component being a chromosome and having the form described above.

See Also

`check_pedigree()`, `sim_ril_pedigree()`, `sim_ail_pedigree()`, `sim_from_pedigree_allchr()`

Examples

```
# simulate AIL pedigree
tab <- sim_ail_pedigree(12, 30)
# simulate data from that pedigree
dat <- sim_from_pedigree(tab)
# simulate multiple chromosomes
dat <- sim_from_pedigree(tab, c("1"=100, "2"=75, "X"=100), xchr="X")
```

---

sim_from_pedigree_allchr

Simulate genotypes for pedigree for multiple chromosomes

Description

Simulate genotypes along all chromosomes for a pedigree. This is a wrap up of sim_from_pedigree.

Usage

```
sim_from_pedigree_allchr(
  pedigree,
  map,
  m = 10,
  p = 0,
  obligate_chiasma = FALSE
)
```

Arguments

`pedigree`	Matrix or data frame describing a pedigree, with first four columns being individual ID, mom ID, dad ID, and sex (female as `0`, male as `1`).
`map`	marker locations, a list with elements for each chromosome

m	Crossover interference parameter, for chi-square model (m=0 corresponds to no interference).
p	proportion of crossovers coming from no-interference process
obligate_chiasma	If TRUE, require an obligate chiasma on the 4-strand bundle at meiosis.

Value

A list with each component being the result from sim_from_pedigree, of length same as map.

See Also

check_pedigree(), sim_ril_pedigree(), sim_ail_pedigree() sim_from_pedigree()

Examples

```
library(qtl)
# marker map
map <- sim.map(len=rep(100, 19), n.mar=10, include.x=FALSE)
# simulate AIL pedigree
tab <- sim_ail_pedigree(12, 30)
# simulate data from that pedigree
dat <- sim_from_pedigree_allchr(tab, map)
```

---

sim_meiosis *Simulate meiosis*

Description

Output a random meiotic product from an input individual.

Usage

```
sim_meiosis(parent, m = 10, p = 0, obligate_chiasma = FALSE, Lstar = NULL)
```

Arguments

parent	An individual object, as output by create_parent() or cross()
m	interference parameter for chi-square model
p	Proportion of chiasmata coming from no-interference process.
obligate_chiasma	If TRUE, require an obligate chiasma on the 4-strand bundle at meiosis.
Lstar	Adjusted chromosome length, if obligate_chiasma=TRUE. Calculated if not provided.

Details

Simulations are under the Stahl model with the interference parameter being an integer. This is an extension of the chi-square model, but with chiasmata being the superposition of two processes, one following the chi-square model and the other exhibiting no interference.

Value

A list with alleles in chromosome intervals (as integers) and locations of the right endpoints of those intervals.

References

Copenhaver, G. P., Housworth, E. A. and Stahl, F. W. (2002) Crossover interference in arabidopsis. *Genetics* **160**, 1631–1639.

Foss, E., Lande, R., Stahl, F. W. and Steinberg, C. M. (1993) Chiasma interference as a function of genetic distance. *Genetics* **133**, 681–691.

Zhao, H., Speed, T. P. and McPeek, M. S. (1995) Statistical analysis of crossover interference using the chi-square model. *Genetics* **139**, 1045–1056.

See Also

`create_parent()`, `cross()`, `sim_crossovers()`, `calc_Lstar()`

Examples

```
ind <- create_parent(100, 1:2)
prod <- sim_meiosis(ind)
```

---

sim_ril_pedigree *Generate a ril pedigree*

Description

Generate a pedigree for multi-way recombinant inbred lines (a table of individual, mom, dad, sex)

Usage

```
sim_ril_pedigree(
  ngen = 20,
  selfing = FALSE,
  parents = 1:2,
  firstind = max(parents) + 1
)
```

Arguments

ngen	Number of generations of inbreeding
selfing	If TRUE, use selfing
parents	Vector of the parents' IDs. Should be integers, and length must be a power of 2 (i.e., 2, 4, 8, ...)
firstind	Positive integer to assign to the first child. Must be greater than max(parents).

Value

A data frame with five columns: individual ID, mother ID, father ID, sex, and generation. Founders have 0 for mother and father ID. Sex is coded 0 for female and 1 for male.

See Also

sim_from_pedigree(), sim_ail_pedigree(), sim_do_pedigree(), sim_4way_pedigree()

Examples

```
tab <- sim_ril_pedigree(7)
```

---

where_het *Find heterozygous regions*

Description

Find regions of heterozygosity in an individual

Usage

```
where_het(ind)
```

Arguments

ind	An individual object, as output be create_parent() or cross()

Value

A matrix with two columns; each row indicates the start and end of a region where the individual is heterozygous

See Also

sim_from_pedigree(), convert2geno()

Examples

```
mom <- create_parent(100, 1:2)
dad <- create_parent(100, 1:2)
child <- cross(mom, dad)
where_het(child)
```

Index